```
FFFFFFFFFFFFFF          333333333
FFFFFFFFFFFFFFF         333333333
FFFFFFFFFFFFFFF         333333333
FFF                 333         333
FFF                 333         333
FFF                 333         333
FFF                             333
FFF                             333
FFF                             333
FFFFFFFFFFFF                333
FFFFFFFFFFFF                333
FFFFFFFFFFFF                333
FFF                             333
FFF                             333
FFF                             333
FFF                 333         333
FFF                 333         333
FFF                 333         333
FFF                     333333333
FFF                   333333333
FFF                   333333333
```

```
LPTSPL VERSION 6(344)  RUNNING ON LPT0
 *START* USER MITS  [6000,6000] JOB  73 SEQ. 42 DATE 10-SEP-75 08:24:32 MONITOR ALBUQUERQUE SCHOOLS 5078 *START*
REQUEST CREATED: 10-SEP-75  03:16:06
FILE: DSK0:F316000,A0001 CREATED: 10-SEP-75 03:16:00 PRINTED: 10-SEP-75 08:52:43
QUEUE SWITCHES:  /FILE:ASCII /COPIES:2 /SPACING:1 /LIMIT:1505 /FORMS:NORMAL
FILE WILL BE DELETED AFTER PRINTING
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(115)  03:12  10-SEP-75  PAGE 1
C       6-SEP-76  03:08          COMMON FILE

     1                                  00100  SEARCH  MC5808                  ;THE UNIVERSAL FILE
     2                                  00200  SUBTTL  COMMON FILE
     3                                  00300  SALL
     4                  000002          00400  LENGTH==2                      ; 0 MEANS 4K, 1 MEANS 8K, 2 MEANS 12K
     5                  000001          00500  REALIO==1
     6                  000001          00600  CASSW==0                       ;CASSETTE SWITCH (CSAVE,CLOAD)
     7                  000000          00700  PURE==0                        ;ON FOR PURE CODE
     8                  000000          00800  LPTSW==0
     9                  000000          00900  DSKFUN==0                      ;ON TO READ/WRITE
    10                  000000          01000  CONSS==0
    11
    12                  000016          01200  CLM=IO=="016                   ;MAKE COMMA COLUMNS FOURTEEN CHARACTERS
    13                  020000          01300  RAMBOT="020000                 ;BOTTOM LOCATION OF RAM FOR PURE SWITCH
    14                  000001          01400  CONTR==1                       ;ALLOW ^O
    15                                  01500  IFE     REALIO,<
    16                                  01600          LPTS==0                ;SIMULATOR DEFAULTS
    17                                  01700          CASS==0
    18                                  01800          CONSS==0
    19                                  01900          DSKFUN==0
    20                                  02000          CONTR==0>
    21
    22                                  02200  IFE     LENGTH,<
    23                                  02300          EXTFNC==0              ;ON MEANS EXTENDED FUNCTIONS
    24                                  02400          MULDIM==0              ;ON MEANS MULTIPLE DIMENSIONED ARRAYS ALLOWED
    25                                  02500          STRING==0              ;ON MEANS STRINGS ALLOWED
    26                                  02600          CASS==0
    27                                  02700          LPTS==0
    28                                  02800          DSFFUN==0
    29                                  02900          CONSS==0
    30                                  03000          CONTR==0>
    31
    32                                  03200  IFE     LENGTH-1,<
    33                                  03300          EXTFNC==1
    34                                  03400          MULDIM==1
    35                                  03500          STRING==1>
    36
    37                                  03700  IFE     LENGTH-2,<
    38                  000001          03800          EXTFNC==1
    39                  000001          03900          MULDIM==1
    40                  000001          04000          STRING==1>
    41
    42                                  04200  DEFINE  SYNCHA(A),<RST  1
    43                                  04300                    A>
    44                                  04400  DEFINE  CHRGET,<RST     2>
    45                                  04500  DEFINE  OUTCHR,<RST     3>
    46                                  04600  DEFINE  COMPAR,<RST     4>
    47                                  04700  DEFINE  FSIGN,<RST      5>
    48                                  04800  DEFINE  PUSHM,<RST      6>
    49                                  04900  DEFINE  PUSHFM,<PUSHM
    50                                  05000                  PUSHM>
    51                                  05100  DEFINE  ACRLF,<
    52                                  05200          "013
    53                                  05300  IFN     STRING,<"010>>
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(113) 03:12 10-SEP-75 PAGE 1-1
C       6-SEP-64 03:06          COMMON FILE

    54                                  05400   DEFINE  PUSHR,<
    55                                  05500           PUSH    D
    56                                  05600           PUSH    B>
    57                                  05700   DEFINE  POPR,<
    58                                  05800           POP     B
    59                                  05900           POP     D>
    60                                  06000   DEFINE  MOVRI(B,C,D,E),<
    61                                  06100           XWD     ^O1000,^O001    ;; "LXI B"
    62                                  06200           EXP     C
    63                                  06300           EXP     B
    64                                  06400           XWD     ^O1000,^O021    ;; "LXI D"
    65                                  06500           EXP     E
    66                                  06600           EXP     D>
    67
    68                                  06800   IF1,<
    69                                  06900   IFE     LENGTH,<PRINTX  /SMALL/>
    70                                  07000   IFE     LENGTH-1,<PRINTX        /MEDIUM/>
    71                                  07100   IFE     LENGTH-2,<PRINTX        /BIG/>
    72                                  07200   IFE     REALIO,<PRINTX  /SIMULATE/>
    73                                  07300   IFN     REALIO,<PRINTX  /ON MACHINE/>
    74                                  07400   IFN     CASSW,<PRINTX   /CASSETTE/>
    75                                  07500   IFN     PURE,<PRINTX    /PURE/>
    76                                  07600   IFN     LPTSW,<PRINTX   /LPT/>
    77                                  07700   IFN     DSKFUN,<PRINTX  /DISK/>
    78                                  07800   IFN     CONSSW,<PRINTX  /CONSOLE/>>
    79                                  07900   PAGE
```

```
BASIC MCS 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(113) 03:12 10-SEP-75 PAGE 1
F3      MAC     6-SEP-64 03:11          VERSION 3.0 -- MORE FEATURES TO GO

    80                                  00020   SUBTTL  VERSION 3.0 -- MORE FEATURES TO GO
    81                                  00040   TITLE   BASIC MCS 8080  GATES/ALLEN/DAVIDOFF
    82                                  00060   IFNDEF  LENGTH,<PRINTX !!! MUST HAVE COM !!!
    83                                  00080           END>
    84  000000'                         00100   MESSIM(START)
    85                                  00120   COMMENT *
    86
    87                                  00160   ********* ____ __ ____ _____ ___ ____ _____
    88                                  00180   COPYRIGHT 1975 BY BILL GATES AND PAUL ALLEN
    89                                  00200   ********* ____ __ ____ _____ ___ ____ _____
    90
    91
    92                                  00260   ORIGINALLY WRITTEN ON THE POP-10 FROM
    93                                  00280   FEBRUARY 9 TO  APRIL 9
    94
    95                                  00320   BILL GATES WROTE THE RUNTIME STUFF,
    96                                  00340   PAUL ALLEN WROTE THE NON-RUNTIME STUFF,
    97                                  00360   MONTE DAVIDOFF WROTE THE MATH PACKAGE.
    98
    99                                  00400   THINGS TO DO:
   100                                  00420   GOSUB / INPUT BUG  (BUF SMASH)
   101                                  00440   PRINT PUNCTUATION MANDATORY
   102                                  00460   MULTIPLE LET
   103                                  00480   RESTART AT 0 SHOULD  GO THROUGH  STKINI
   104                                  00500   USER DEFINED FUNCTIONS(MULTI-ARG,MULTI-LINE,STRINGS)
   105                                  00520   MAKE STACK BOUNDARY STUFF EXACT
   106                                  00540   PUNCH,RENUMBER,,,
   107                                  00560   INLINE CONSTANT CONVERSION--MAKE IT WORK
   108                                  00580   *
   109                                  00600   RADIX 10                        ;THROUGHOUT THE NON-MATH PACKAGE ROUTINES
   110             000000               00620   .P==0                           ;FOR SIMULATOR FIXUPS
   111             000025               00640   NUMLEV==17+LENGTH*2             ;NUMBER OF STACK LEVELS RESERVED
   112                                  00660                                   ;WITH AN EXPLICIT CALL TO GETSTK
   113             000110               00680   LPTLEN==72                      ;WIDTH OF LINE-PRINTER
   114             000110               00700   LINLEN==72                      ;TELETYPE LINE LENGTH
   115             000110               00720   BUFLEN==72                      ;INPUT BUFFER SIZE
   116             000004               00740   STRSIZ==4
   117             000003               00760   IFE     LENGTH-2,<STRSIZ==3>
   118             000003               00780   NUMTMP==3                       ;NUMBER OF STRING TEMPORARIES
   119             000005               00800   IFE     LENGTH-2,<NUMTMP=5>
   120             000015               00820   CR==13
   121             000017               00840   CONT==15                        ;CHARACTER TO SUPRESS OUTPUT
   122             000200               00860   DODNE==128                      ;BIT FOR OUTPUT DONE
   123             000001               00880   IODNE==1                        ;BIT FOR INPUT DONE
   124             000001               00900   TTOCHN==1                       ;TELETYPE OUTPUT CHANNEL
   125             000001               00920   TTICHN==1                       ;TELETYPE INPUT CHANNEL
   126             000000               00940   LISTEN==0                       ;ON MEANS LISTEN FOR ^C
   127             000001               01000   FUNCTS==1                       ;ON MEANS USER FUNCTIONS ALLOWED
   128                                  01020                                   ;; CANNOT BE ON WITH LENGTH=0 (SUBFLG)
   129                                  01040   IFN     REALIO,<
   130             000001               01060   LISTEN==1>
   131                                  01080   IFE     LENGTH,<
   132                                  01120   FUNCTS==0>
```

```
BASIC MC8 8080  GATES/ALLEN/DAVIDOFF    MACRO 4/(113) 03:12 10-SEP-75 PAGE 1-1
F5      MAC     6-SEP-80 03:11          VERSION 5.0 -- MORE FEATURES TO GO

   133
   134
   135                          01200   INTERNAL        ,C1,BUF,READY,REASON,SNERR,OMERR,KEPINI
   136                          01220   INTERNAL        STREND,CURLIN,DV0ERR,ERROV
   137                          01240   IFN     REALIO,<
   138                          01260   INTERNAL        CNLCA1,CNLCA2,CNLCA3>
   139                          01280   IFN     EXTFNC,<INTERNAL        ATNFIX,COSFIX,SINFIX,TANFIX
   140                          01300   EXTERNAL        FPWR,EXP>
   141                          01320   EXTERNAL        QINT,ZERO,MOVE,FOUT,FIN,FCOMP,FADD,PUSHF,INT,INIT
   142                          01340   EXTERNAL        MOVFR,MOVRF,MOVRM,INPRT,LINPRT
   143                          01360   EXTERNAL        MOVFM,MOVMF,TSTACK,FLOATRI,FADDS
   144                          01380   INTERNAL        ILLFUN,FAC,FACLO,TXTTAB,STROUT,SCRTCH
   145                          01400   EXTERNAL        INPART,NEG,FLOAT
   146                          01420   INTERNAL        OUTDO,STRQUI
   147                          01440   INTERNAL        STKTOP,ERROR,FCERR
   148                          01460   IFN     STRING,<
   149                          01462   INTERNAL        VALTYP,TEMPPT
   150                          01464   INTERNAL        TEMPST,STRLIT
   151                          01466   IFN     LENGTH-2,<
   152                          01468   INTERNAL        TMERR>
   153                          01480   INTERNAL        MEMSIZ,FRETOP
   154                          01500   EXTERNAL        SIGNS>
   155                          01520   INTERNAL        FBUFFR,MINUTE,PLUSTK,CRDO,LINGET,INTXT,GINLIN
   156                          01540   IFN     MULDIM,<INTERNAL        BSERR>
   157                          01560   IFN     MULDIM,<EXTERNAL        DMULT>
   158                          01580   IFE     LENGTH,<INTERNAL        RNDFIX,SGNFIX,SINFIX>
   159                          01600   INTERNAL        SIGN
   160                          01620   EXTERNAL        SIGNC,POPHRT
   161                          01640   IFN     CONTRW,<
   162                          01660   INTERNAL        CNTNFL>
   163                          01680   IFN     LPTSW,<
   164                          01700           INTERNAL        LPTPOS,PRTFLG>
   165                          01720   IFE     LENGTH-2,<
   166                          01730   EXTERNAL        CONSIN,VMOVFA,VMOVAF,ISIGN,FPWRQ,CONIA,GETBCD,VSIGN
   167                          01740   EXTERNAL        VMOVMF,VMOVFM,FRCINT,FRCSNG,FRCDBL,VNEG,PUFOUT,DCXBRT,DADD
   168                          01760   EXTERNAL        DSUB,DMULT,DDIV,DCOMP,INEG,DABS,DSUB,DMULT,DDIV,DCOMP,VINT
   169                          01780   EXTERNAL        TMERR,VMOVE,VALINT,VALSNG,FRCSTR,CHKSTR,MAKINT
   170                          01800   INTERNAL        DFACLO,ARG,ARGLO,VALTYP,ERRTM,TEMP2,TEMP3,GETYPE>
   171                          01820   PAGE


BASIC MC8 8080  GATES/ALLEN/DAVIDOFF    MACRO 4/(113) 03:12 10-SEP-75 PAGE 2
F5      MAC     6-SEP-80 03:11          SOME EXPLANATION

   172                          01840   SUBTTL  SOME EXPLANATION
   173
   174                          01880   COMMENT *
   175
   176                          01920   ALTAIR BASIC CONFIGURES MEMORY AS FOLLOWS:
   177
   178                          01960   LOW LOCATIONS
   179
   180                          02000           RST     SUBROUTINES
   181
   182                          02040                   0       STARTUP
   183                          02060                           INITIALLY A JMP TO THE INITIALIZATION CODE
   184                          02080                           BUT CHANGED TO A JMP TO READY,
   185                          02100                           RESTARTING THE MACHINE AT 0 DURING PROGRAM
   186                          02120                           EXECUTION CAN LEAVE THINGS MESSED UP.
   187
   188                          02160                   1       SYNCHK
   189                          02180                           A CHECK IS MADE TO MAKE SURE THE
   190                          02200                           CHARACTER POINTER POINTS AT A SPECIFIC
   191                          02220                           CHARACTER, IF NOT THE "SYNTAX ERROR"
   192                          02240                           ROUTINE IS CALLED. IF SO,
   193                          02260                           THE CHRGET RST IS DROPPED INTO SO
   194                          02280                           THE CHARACTER AFTER THE MATCHED
   195                          02300                           ONE WILL BE PUT IN [A] AND
   196                          02320                           THE CONDITION CODES WILL REFLECT THIS
   197                          02340                           EXAMPLE: SYNCHK THENTA (THE MATCH CHARACTER IS
   198                          02360                           GIVEN IN THE LOCATION AFTER THE RST)
   199                          02380                           WOULD CHECK TO MAKE SURE (H,L) POINTED TO A THENTA
   200                          02400                           AND IF SO FETCH THE NEXT CHARACTER INTO [A].
   201                          02420                           IF NOT, A "SYNTAX ERROR" WOULD BE GIVEN.
   202
   203                          02460                   2       CHRGET
   204                          02480                           USING (H,L) AS THE TEXT POINTER
   205                          02500                           THE TEXT POINTER IS INCREMENTED
   206                          02520                           AND THE NEXT CHARACTER IS FETCHED INTO [A]
   207                          02540                           IF THE CHARACTER IS A " " IT IS SKIPPED
   208                          02560                           OVER AND THE NEXT CHARACTER IS FETCHED.
   209                          02580                           THE STATEMENT TERMINATORS ":" AND 0
   210                          02600                           LEAVE THE ZERO FLAG SET,
   211                          02620                           THE NUMERICS "0" THROUGH "9" LEAVE THE CARRY
   212                          02640                           FLAG SET, THE CURRENT CHARACTER CAN BE
   213                          02660                           REFETCHED INTO [A] BY DOING A MOV A,M,
   214                          02680                           IF THE CONDITION CODES MUST BE SET UP AGAIN
   215                          02700                           DCX H,CHRGET WILL WORK. IT IS VERY DIFFICULT
   216                          02720                           TO REEXAMINE THE CHARACTER BEFORE THE CURRENT
   217                          02740                           ONE SINCE SPACES MAY BE IN-BETWEEN.
   218                          02760                           DCX H,DCX H,CHRGET WILL NOT ALWAYS WORK.
   219
   220                          02800                   3       OUTCHR
   221                          02820                           THE CHARACTER IN [A] IS PRINTED ON
   222                          02840                           THE USER'S TERMINAL. [A] AND THE
   223                          02860                           CONDITION CODES ARE PRESERVED.
   224
```

```
                                        4       COMPAR
                                                (D,E) AND (H,L) ARE COMPARED AS UNSIGNED
                                                DOUBLE-BYTE INTEGERS, CARRY IS SET IF
                                                (H,L) IS LESS THAN (D,E), ZERO IS SET IF THEY
                                                ARE EQUAL. (A) IS SMASHED, THE ONLY DEFINITE
                                                THING THAT CAN BE SAID ABOUT (A) ON RETURN
                                                IS THAT IF THE ZERO FLAG IS SET, (A) WILL
                                                EQUAL 0.

                                        5       FSIGN
                                                THE FAC (FLOATING ACCUMULATOR)
                                                WHICH IS USED TO STORE NUMERIC RESULTS
                                                IS CHECKED TO SEE WHAT SIGN ITS
                                                VALUE HAS.

                                        6       PUSHH
                                                A DOUBLE BYTE QUANTITY POINTED
                                                TO BY (H,L) IS PUSHED ONTO THE
                                                STACK. (B,C) IS SET EQUAL TO THE
                                                VALUE PUSHED. (H,L) IS INCREMENTED BY TWO.

                                        7       IN THE 4K VERSION RST 7 IS UNUSED AND THE LOCATIONS
                                                ASSOCIATED WITH IT ARE USED TO CONTINUE
                                                THE CODE FOR RST 6. IN THE 8K A JMP IS MADE
                                                AROUND THE FIRST THREE RST 7 LOCATIONS
                                                DURING RST 6 EXECUTION. RST 7 INITIALLY
                                                CONTAINS A RET, BUT THE USER CAN CHANGE IT TO
                                                A JMP TO AN INTERRUPT SERVICE ROUTINE.

                                FUNCTION DISPATCH ADDRESSES
                                        FUNDSP CONTAINS THE ADDRESSES OF THE
                                        FUNCTION ROUTINES IN THE ORDER OF THE
                                        FUNCTION NAMES IN THE CRUNCH LIST.
                                        THE FUNCTIONS THAT TAKE MORE THAN ONE ARGUMENT
                                        ARE AT THE END. SEE THE EXPLANATION AT ISFUN.

                                THE OPERATOR TABLE
                                        THE OPTAB TABLE CONTAINS AN OPERATORS PRECEDENCE
                                        FOLLOWED BY THE ADDRESS OF THE ROUTINE TO PERFORM
                                        THE OPERATION. THE INDEX INTO THE
                                        OPERATOR TABLE IS MADE BY SUBTRACTING OFF THE CRUNCH VALUE
                                        OF THE LOWEST NUMBERED OPERATOR. THE ORDER
                                        OF OPERATORS IN THE CRUNCH LIST AND IN OPTAB IS IDENTICAL.
                                        THE PRECEDENCES ARE ARBITRARY,EXCEPT FOR THEIR
                                        COMPARATIVE SIZES. NOTE THAT THE PRECEDENCE FOR
                                        UNARY OPERATORS SUCH AS NOT AND NEGATION ARE
                                        SETUP SPECIALLY WITHOUT USING A TABLE.

                                THE RESERVED WORD OR CRUNCH LIST
                                        WHEN A COMMAND OR PROGRAM LINE IS TYPED IN
                                        IT IS STORED IN BUF. AS SOON AS THE WHOLE LINE
                                        HAS BEEN TYPED IN (INLIN RETURNS) CRUNCH IS
                                        CALLED TO CONVERT ALL RESERVED WORDS TO THEIR
```



```
                                        CRUNCH VALUES. THIS REDUCES THE SIZE OF THE
                                        PROGRAM AND SPEEDS UP EXECUTION BY ALLOWING
                                        TABLE DISPATCHES TO PERFORM FUNCTIONS,STATEMENTS,
                                        AND OPERATIONS. THIS IS BECAUSE ALL THE STATEMENT
                                        NAMES ARE STORED CONSECUTIVELY IN THE CRUNCH LIST,
                                        WHEN A MATCH IS FOUND BETWEEN A STRING
                                        OF CHARACTERS AND A WORD IN THE CRUNCH LIST
                                        THE ENTIRE TEXT OF THE MATCHED WORD IS TAKEN OUT OF
                                        THE INPUT LINE AND A RESERVED WORD TOKEN IS PUT
                                        IN ITS PLACE. A RESERVED WORD TOKEN IS ALWAYS EQUAL
                                        TO OCTAL 200 PLUS THE POSITION OF THE MATCHED WORD
                                        IN THE CRUNCH LIST.

                                STATEMENT DISPATCH ADDRESSES
                                        WHEN A STATEMENT IS TO BE EXECUTED, THE FIRST
                                        CHARACTER OF THE STATEMENT IS EXAMINED
                                        TO SEE IF IT IS LESS THAN THE RESERVED
                                        WORD TOKEN FOR THE LOWEST NUMBERED STATEMENT NAME.
                                        IF SO, THE "LET" CODE IS CALLED TO
                                        TREAT THE STATEMENT AS AN ASSIGNMENT STATEMENT.
                                        OTHERWISE A CHECK IS MADE TO MAKE SURE THE
                                        RESERVED WORD NUMBER IS NOT TOO LARGE TO BE A
                                        STATEMENT TYPE NUMBER. IF NOT THE ADDRESS
                                        TO DISPATCH TO IS FETCHED FROM STMDSP (THE STATEMENT
                                        DISPATCH TABLE) USING THE RESERVED WORD
                                        NUMBER FOR THE STATEMENT TO CALCULATE AN INDEX INTO
                                        THE TABLE.

                                ERROR MESSAGES
                                        WHEN AN ERROR CONDITION IS DETECTED
                                        (E) MUST BE SET UP TO INDICATE WHICH ERROR
                                        MESSAGE IS APPROPRIATE AND A BRANCH MUST BE MADE
                                        TO ERROR. THE STACK WILL BE RESET AND ALL
                                        PROGRAM CONTEXT WILL BE LOST. VARIABLES
                                        VALUES AND THE ACTUAL PROGRAM REMAIN INTACT.
                                        ONLY THE VALUE OF (E) IS IMPORTANT WHEN
                                        THE BRANCH IS MADE TO ERROR. (E) IS USED AS AN
                                        INDEX INTO ERRTAB WHICH GIVES THE TWO
                                        CHARACTER ERROR MESSAGE THAT WILL BE PRINTED ON THE
                                        USER'S TERMINAL.

                                IMPURE STORAGE
                                        ALL TEMPORARIES,FLAGS,POINTERS, THE BUFFER AREA,
                                        THE FLOATING ACCUMULATOR,AND ANYTHING ELSE THAT
                                        IS USED TO STORE A CHANGING VALUE SHOULD BE LOCATED
                                        IN THIS AREA. CARE MUST BE MADE IN MOVING LOCATIONS
                                        IN THIS AREA SINCE THE JUXTAPOSITION OF TWO LOCATIONS
                                        IS OFTEN DEPENDED UPON.

                                TEXTUAL MESSAGES
                                        CONSTANT MESSAGES ARE STORED HERE, UNLESS
                                        THE CODE TO CHECK IF A STRING MUST BE COPIED
                                        IS CHANGED THESE STRINGS MUST BE STORED ABOVE
```

```
                                05020                   DSCTMP, OR ELSE THEY WILL BE COPIED BEFORE
                                05040                   THEY ARE PRINTED.
                                05080           FNDFOR
                                05100                   MOST SMALL ROUTINES ARE FAIRLY SIMPLE
                                05120                   AND ARE DOCUMENTED IN PLACE, FNDFOR IS
                                05140                   USED FOR FINDING "FOR" ENTRIES ON
                                05160                   THE STACK, WHENEVER A "FOR" IS EXECUTED AN
                                05180                   18 BYTE ENTRY IS PUSHED ONTO THE STACK.
                                05200                   BEFORE THIS IS DONE, HOWEVER, A CHECK
                                05220                   MUST BE MADE TO SEE IF THERE
                                05240                   ARE ANY "FOR" ENTRIES ALREADY ON THE STACK
                                05260                   FOR THE SAME LOOP VARIABLE, IF SO, THAT "FOR" ENTRY
                                05280                   AND ALL OTHER "FOR" ENTRIES THAT WERE MADE AFTER IT
                                05300                   ARE ELIMINATED FROM THE STACK, THIS IS SO A
                                05320                   PROGRAM THAT JUMPS OUT OF THE MIDDLE
                                05340                   OF A "FOR" LOOP AND THEN RESTARTS THE LOOP AGAIN
                                05360                   AND AGAIN WON'T USE UP 18 BYTES OF STACK
                                05380                   SPACE EVERY TIME, THE "NEXT" CODE ALSO
                                05400                   CALLS FNDFOR TO SEARCH FOR A "FOR" ENTRY WITH
                                05420                   THE LOOP VARIABLE IN
                                05440                   THE "NEXT", AT WHATEVER POINT A MATCH IS FOUND
                                05460                   THE STACK IS RESET, IF NO MATCH IS FOUND A
                                05480                   "NEXT WITHOUT FOR"  ERROR OCCURS, GOSUB EXECUTION
                                05500                   ALSO PUTS A 6 BYTE ENTRY ON STACK,
                                05520                   WHEN A RETURN IS EXECUTED FNDFOR IS
                                05540                   CALLED WITH A VARIABLE POINTER THAT CAN'T
                                05560                   BE MATCHED, WHEN "FNDFOR" HAS RUN
                                05580                   THROUGH ALL THE "FOR" ENTRIES ON THE STACK
                                05600                   IT RETURNS AND THE RETURN CODE MAKES
                                05620                   SURE THE ENTRY THAT WAS STOPPED
                                05640                   ON IS A GOSUB ENTRY, THIS ASSURES THAT
                                05660                   IF YOU GOSUB TO A SECTION OF CODE
                                05680                   IN WHICH A FOR LOOP IS ENTERED BUT NEVER
                                05700                   EXITED THE RETURN WILL STILL BE
                                05720                   ABLE TO FIND THE MOST RECENT
                                05740                   GOSUB ENTRY, THE "RETURN" CODE ELIMINATES THE
                                05760                   "GOSUB" ENTRY AND ALL "FOR" ENTRIES MADE AFTER
                                05780                   THE GOSUB ENTRY.
                                05820           NON-RUNTIME STUFF
                                05840                   THE CODE TO INPUT A LINE,CRUNCH IT,GIVE ERRORS,
                                05860                   FIND A SPECIFIC LINE IN THE PROGRAM,
                                05880                   PERFORM A "NEW","CLEAR", AND "LIST" ARE
                                05900                   ALL IN THIS AREA, GIVEN THE EXPLANATION OF
                                05920                   PROGRAM STORAGE GIVEN BELOW THESE ARE
                                05940                   ALL STRAIGHTFORWARD.
                                05980           NEWSTT
                                06000                   WHENEVER A STATEMENT FINISHES EXECUTION IT
                                06020                   DOES A "RET" WHICH TAKES
                                06040                   EXECUTION BACK TO NEWSTT, STATEMENTS THAT
                                06060                   CREATE OR LOOK AT SEMI-PERMANENT STACK ENTRIES
```



```
                                06080                   MUST GET RID OF THE RETURN ADDRESS OF NEWSTT AND
                                06100                   JMP TO NEWSTT WHEN DONE, NEWSTT ALWAYS
                                06120                   CHARGETS THE FIRST CHARACTER AFTER THE STATEMENT
                                06140                   NAME BEFORE DISPATCHING, WHEN RETURNING
                                06160                   BACK TO NEWSTT THE ONLY THING THAT
                                06180                   MUST BE SET UP IS THE TEXT POINTER IN
                                06200                   (H,L), NEWSTT WILL CHECK TO MAKE SURE
                                06220                   (H,L) IS POINTING TO A STATEMENT TERMINATOR,
                                06240                   IF A STATEMENT SHOULDN'T BE PERFORMED UNLESS
                                06260                   IT IS PROPERLY FORMATTED (I.E. "NEW") IT CAN
                                06280                   SIMPLY DO A "RNZ" AFTER READING ALL OF
                                06300                   ITS ARGUMENTS, SINCE THE ZERO FLAG
                                06320                   BEING OFF INDICATES THERE IS NOT
                                06340                   A STATEMENT TERMINATOR NEWSTT WILL
                                06360                   DO THE JMP TO THE "SYNTAX ERROR"
                                06380                   ROUTINE, IF A STATEMENT SHOULD BE STARTED
                                06400                   OVER IT CAN DO LHLD TEMP,RET SINCE THE (H,L)
                                06420                   AT NEWSTT IS ALWAYS STORED IN TEMP, OF COURSE
                                06440                   CARE MUST BE TAKEN THAT NO ROUTINE
                                06460                   THAT SMASHES TEMP HAS BEEN CALLED,
                                06480                   THE "C CODE STORES TEMP IN OLDTXT AND CURLIN (THE
                                06500                   CURRENT LINE NUMBER) IN OLDLIN SINCE THE "C CHECK
                                06520                   IS MADE BEFORE THE STATEMENT POINTED TO IS
                                06540                   EXECUTED, "STOP" AND "END" STORE THE TEXT POINTER
                                06560                   IN (H,L) WHICH POINTS AT THEIR TERMINATING
                                06580                   CHARACTER IN OLDTXT,
                                06620           STATEMENT CODE
                                06640                   THE INDIVIDUAL STATEMENT CODE COMES
                                06660                   NEXT, THE APPROACH USED IN EXECUTING EACH
                                06680                   STATEMENT IS DOCUMENTED IN THE STATEMENT CODE
                                06700                   ITSELF,
                                06740           FRMEVL, THE FORMULA EVALUATOR
                                06760                   GIVEN AN (H,L) POINTING TO THE STARTING
                                06780                   CHARACTER OF A FORMULA FRMEVL
                                06800                   EVALUATES THE FORMULA AND LEAVES
                                06820                   THE VALUE IN THE FLOATING ACCUMULATOR (FAC),
                                06840                   (H,L) IS RETURNED POINTING TO THE FIRST CHARACTER
                                06860                   THAT COULD NOT BE INTERPRETED AS PART OF THE
                                06880                   FORMULA, THE ALGORITHM USES THE STACK
                                06900                   TO STORE TEMPORARY RESULTS:
                                06940                           0, PUT A DUMMY PRECEDENCE OF ZERO ON
                                06960                                   THE STACK.
                                07000                           1, READ LEXEME (CONSTANT,FUNCTION,
                                07020                                   VARIABLE,FORMULA IN PARENS)
                                07040                                   AND TAKE THE LAST PRECEDENCE VALUE
                                07060                                   OFF THE STACK,
                                07080                           2, SEE IF THE NEXT CHARACTER IS AN OPERATOR
                                07100                                   IF NOT,RETURN, THIS MAY CAUSE
                                07120                                   OPERATOR APPLICATION OR AN ACTUAL
                                                                        RETURN FROM FRMEVL.
```

```
                                                07140                           3, IF IT IS, SEE WHAT PRECEDENCE IT HAS
                                                07160                              AND COMPARE IT TO THE PRECEDENCE
                                                07180                              OF THE LAST OPERATOR ON THE STACK
                                                07200                           4, IF = OR LESS REMEMBER THE TEXT
                                                07220                              POINTER AT THE START OF THIS OPERATOR
                                                07240                              AND DO A RETURN TO CAUSE
                                                07260                              APPLICATION OF THE LAST OPERATOR,
                                                07280                              EVENTUALLY RETURN TO STEP 2
                                                07300                              BY RETURNING TO RETADP,
                                                07320                           5, IF GREATER PUT THE LAST PRECEDENCE
                                                07340                              BACK ON, SAVE THE CURRENT
                                                07360                              TEMPORARY RESULT, OPERATOR ADDRESS
                                                07380                              AND PRECEDENCE AND RETURN TO STEP 1,

                                                07420                   RELATIONAL OPERATORS ARE ALL HANDLED THROUGH
                                                07440                   A COMMON ROUTINE, SPECIAL
                                                07460                   CARE IS TAKEN TO DETECT TYPE MISMATCHES SUCH AS 3="F"

                                                07500           EVAL -- THE ROUTINE TO READ A LEXEME
                                                07520                   EVAL CHECKS FOR THE DIFFERENT TYPES OF
                                                07540                   ENTITIES IT IS SUPPOSED TO DETECT,
                                                07560                   LEADING PLUSES ARE IGNORED,
                                                07580                   DIGITS AND "." CAUSE FIN (FLOATING INPUT)
                                                07600                   TO BE CALLED, FUNCTION NAMES CAUSE THE
                                                07620                   FORMULA INSIDE THE PARENTHESES TO BE EVALUATED
                                                07640                   AND THE FUNCTION ROUTINE TO BE CALLED, VARIABLE
                                                07660                   NAMES CAUSE PTRGET TO BE CALLED TO GET A POINTER
                                                07680                   TO THE VALUE, AND THEN THE VALUE IS PUT INTO
                                                07700                   THE FAC, AN OPEN PARENTHESIS CAUSES FRMEVL
                                                07720                   TO BE CALLED (RECURSIVELY), AND THE ")" TO
                                                07740                   BE CHECKED FOR, UNARY OPERATORS (NOT AND
                                                07760                   NEGATION)  PUT THEIR PRECEDENCE ON THE STACK
                                                07780                   AND ENTER FORMULA EVALUATION AT STEP 1, SO
                                                07800                   THAT EVERYTHING UP TO AN OPERATOR GREATER THAN
                                                07820                   THEIR PRECEDENCE OR THE END OF THE FORMULA
                                                07840                   WILL BE EVALUATED, WHEN FRMEVL DOES A RETURN
                                                07860                   BECAUSE IT SEES AN OPERATOR OF HIGHER PRECEDENCE
                                                07880                   IT DOES NOT PASS THE TEXT POINTER IN H,L), SO
                                                07900                   AFTER THE UNARY OPERATION HAS BEEN PERFORMED
                                                07920                   ON THE FAC THE TEXT POINTER MUST BE FETCHED FROM
                                                07940                   A TEMPORARY LOCATION THAT FRMEVL USES AND
                                                07960                   A RETURN BACK TO FRMEVL DONE,

                                                08000           DIMENSION AND VARIABLE SEARCHING
                                                08020                   SPACE IS ALLOCATED FOR VARIABLES AS THEY ARE
                                                08040                   ENCOUNTERED, THUS "DIM" STATEMENTS MUST BE
                                                08060                   EXECUTED TO HAVE EFFECT, 6 BYTES ARE ALLOCATED
                                                08080                   FOR EACH SIMPLE VARIABLE, WHETHER IT IS A STRING,
                                                08100                   NUMBER OR USER DEFINED FUNCTION, THE FIRST TWO
                                                08120                   BYTES GIVE THE NAME OF THE VARIABLE AND THE LAST FOUR
                                                08140                   GIVE ITS VALUE, (VARTAB) GIVES THE FIRST LOCATION
                                                08160                   WHERE A SIMPLE VARIABLE NAME IS FOUND AND (ARYTAB)
                                                08180                   GIVES THE LOCATION TO STOP SEARCHING FOR SIMPLE
```



```
                                                08200                   VARIABLES, A "FOR" ENTRY HAS A TEXT POINTER
                                                08220                   AND A POINTER TO A VARIABLE VALUE SO NEITHER
                                                08240                   THE PROGRAM OR THE SIMPLE VARIABLES CAN BE
                                                08260                   MOVED WHILE THERE ARE ACTIVE "FOR" ENTRIES ON THE STACK,
                                                08280                   USER DEFINED FUNCTION VALUES ALSO CONTAIN
                                                08300                   POINTERS INTO SIMPLE VARIABLE SPACE SO NO USER-DEFINED
                                                08320                   FUNCTION VALUES CAN BE RETAINED IF SIMPLE VARIABLES
                                                08340                   ARE MOVED, ADDING A SIMPLE VARIABLE
                                                08360                   ADDING SIX TO ARYTAB AND STREND, BLOCK TRANSFERING
                                                08380                   THE ARRAY VARIABLES UP BY SIX AND MAKING SURE THE
                                                08400                   NEW (STREND) IS NOT TO CLOSE TO THE STACK,
                                                08420                   THIS MOVEMENT OF ARRAY VARIABLES MEANS
                                                08440                   THAT NO POINTER TO AN ARRAY WILL STAY VALID WHEN
                                                08460                   NEW SIMPLE VARIABLES CAN BE ENCOUNTERED, THIS IS
                                                08480                   WHY ARRAY VARIABLES ARE NOT ALLOWED "FOR"
                                                08500                   LOOP VARIABLES, SETING UP A NEW ARRAY VARIABLE
                                                08520                   MERELY INVOLVES BUILDING THE DESCRIPTOR,
                                                08540                   UPDATING STREND, AND MAKING SURE THERE IS
                                                08560                   STILL ENOUGH ROOM BETWEEN STREND AND THE
                                                08580                   STACK, WITHOUT MULTIPLE DIMENSIONS THE FORMAT
                                                08600                   OF AN ARRAY VARIABLE IS SIMPLY:
                                                08620                           SECOND CHARACTER
                                                08640                           FIRST CHARACTER
                                                08660                           NUMBER OF BYTES USED BY VALUES
                                                08680                           VALUES
                                                08700                   THE FORMAT WHEN MULTIPLY DIMENSIONED VARIABLES
                                                08720                   ARE ALLOWED IS DESCRIBED IN THE "MULDIM" CODE,
                                                08740                   PTRGET, THE ROUTINE WHICH RETURNS A POINTER
                                                08760                   TO A VARIABLE VALUE, HAS TWO IMPORTANT FLAGS, ONE IS
                                                08780                   "DIMFLG" WHICH INDICATED WHETHER "DIM" CALLED PTRGET
                                                08800                   OR NOT, IF SO, NO PRIOR ENTRY FOR THE VARIABLE IN
                                                08820                   QUESTION SHOULD BE FOUND, AND THE INDEX INDICATES
                                                08840                   HOW MUCH SPACE TO SET ASIDE, SIMPLE VARIABLES CAN
                                                08860                   BE "DIMENSIONED", BUT THE ONLY EFFECT WILL BE TO
                                                08880                   SET ASIDE SPACE FOR THE VARIABLE IF IT HASN'T BEEN
                                                08900                   ENCOUNTERED YET, THE OTHER IMPORTANT FLAG IS SUBFLG
                                                08920                   WHICH INDICATES WHETHER A SUBSCRIPTED VARIABLE SHOULD BE
                                                08940                   ALLOWED IN THE CURRENT CONTEXT, IF SUBFLG IS NON-ZERO
                                                08960                   THE OPEN PARENTHESIS FOR A SUBSCRIPTED VARIABLE
                                                08980                   WILL NOT BE SCANNED BY PTRGET, AND PTRGET WILL RETURN
                                                09000                   WITH A TEXT POINTER POINTING TO THE "(", IF
                                                09020                   THERE WAS ONE,
                                                09040           STRINGS
                                                09060                   IN THE VARIABLE TABLE STRINGS ARE STORED JUST LIKE
                                                09080                   NUMERIC VARIABLES, SIMPLE STRINGS HAVE FOUR VALUE
                                                09100                   BYTES WHICH ARE INITIALIZED TO ALL ZEROS (WHICH
                                                09120                   REPRESENTS THE NULL STRING), THE ONLY DIFFERENCE
                                                09140                   IN HANDLING IS THAT WHEN PTRGET SEES A "$" AFTER THE
                                                09160                   NAME OF A VARIABLE, PTRGET SETS VALTYP TO ONE AND TURNS
                                                09180                   ON THE MSB (MOST-SIGNIFICANT-BIT) OF THE VALUE OF
                                                09200                   THE FIRST CHARACTER OF THE VARIABLE NAME,
                                                09220                   HAVING THIS BIT ON IN THE NAME OF THE VARIABLE ENSURES
                                                09240                   THAT THE SEARCH ROUTINE WILL NOT MATCH
```

```
                        09260
                        09280
                        09300
                        09320
                        09340
                        09360
                        09380
                        09400
                        09420
                        09440
                        09460
                        09480
                        09500
                        09520
                        09540
                        09560
                        09580
                        09600
                        09620
                        09640
                        09660
                        09680
                        09700
                        09720
                        09740
                        09760
                        09780
                        09800
                        09820
                        09840
                        09860
                        09880
                        09900
                        09920
                        09940
                        09960
                        09980
                        10000
                        10020
                        10040
                        10060
                        10080
                        10100
                        10120
                        10140
                        10160
                        10180
                        10200
                        10220
                        10240
                        10260
                        10280
                        10300
```

```
                        'A' WITH 'AS' OR 'AS' WITH 'A', THE MEANING OF
                        THE FOUR VALUE BYTES ARE:
                                LOW
                                        LENGTH OF THE STRING
                                        UNUSED
                                        LOW 8 BITS
                                        HIGH 8 BITS OF THE ADDRESS
                                            OF THE CHARACTERS IN THE
                                            STRING IF LENGTH.NE.0.
                                            MEANINGLESS OTHERWISE.
                                HIGH
                        THE VALUE OF A STRING VARIABLE (THESE 4 BYTES)
                        IS CALLED THE STRING DESCRIPTOR TO DISTINGUISH
                        IT FROM THE ACTUAL STRING DATA, WHENEVER A
                        STRING CONSTANT IS ENCOUNTERED IN A FORMULA OR AS
                        PART OF AN INPUT STRING, OR AS PART OF DATA, STRLIT
                        IS CALLED, CAUSING A DESCRIPTOR TO BE BUILT FOR
                        THE STRING. IF THE STRING CONSTANT IS IN BUF (WHICH
                        IT WILL BE IF THE STRING IS BEING "INPUT", OR THE
                        STRING IS PART OF SOME FORMULA IN A DIRECT STATEMENT)
                        THE VALUE IS COPIED INTO STRING SPACE SINCE BUF
                        IS ALWAYS CHANGING, "STKCPY" IS USED TO COPY
                        STRINGS.

                        STRING FUNCTIONS AND THE ONE STRING OPERATOR "+"
                        ALWAYS RETURN THEIR VALUES IN STRING SPACE,
                        ASSIGNING A STRING A CONSTANT VALUE IN A PROGRAM
                        THROUGH A "READ" OR ASSIGNMENT STATEMENT
                        WILL NOT USE ANY STRING SPACE SINCE
                        THE STRING DESCRIPTOR WILL POINT INTO THE
                        PROGRAM ITSELF. IN GENERAL, COPYING IS DONE
                        WHEN A STRING VALUE IS IN BUF, OR IT IS IN STRING
                        SPACE AND THERE IS AN ACTIVE POINTER TO IT.
                        THUS F$=G$ WILL CAUSE COPYING IF G$ HAS ITS
                        STRING DATA IN STRING SPACE. F$=CHR$(7)
                        WILL USE ONE BYTE OF STRING SPACE TO STORE THE
                        NEW ONE CHARACTER STRING CREATED BY "CHR$", BUT
                        THE ASSIGNMENT ITSELF WILL CAUSE NO COPYING SINCE
                        THE ONLY POINTER AT THE NEW STRING IS A
                        TEMPORARY DESCRIPTOR CREATED BY FRMEVL WHICH WILL
                        GO AWAY AS SOON AS THE ASSIGNMENT IS DONE.
                        IT IS THE NATURE OF GARBAGE COLLECTION THAT
                        DISALLOWS HAVING TWO STRING DESCRIPTORS POINT TO THE SAME
                        AREA IN STRING SPACE, STRING FUNCTIONS AND OPERATORS
                        MUST PROCEED AS FOLLOWS:
                                1) FIGURE OUT THE LENGTH OF THEIR RESULT
                                2) CALL GETSPA TO FIND SPACE FOR THEIR
                                RESULT, THE ARGUMENTS TO THE FUNCTION
                                OR OPERATOR MAY CHANGE SINCE GARBAGE COLLECTION
                                MAY BE INVOKED. THE ONLY THING THAT CAN
                                BE SAVED DURING THE CALL TO GETSPA IS A POINTER
                                TO THE DESCRIPTORS OF THE ARGUMENTS.
                                3) CONSTRUCT THE RESULT DESCRIPTOR IN DSCTMP.
```



```
                        10320
                        10340
                        10360
                        10380
                        10400
                        10420
                        10440
                        10460
                        10480
                        10500
                        10520
                        10540
                        10560
                        10580
                        10600
                        10620
                        10640
                        10660
                        10680
                        10700
                        10720
                        10740
                        10760
                        10780
                        10800
                        10820
                        10840
                        10860
                        10880
                        10900
                        10920
                        10940
                        10960
                        10980
                        11000
                        11020
                        11040
                        11060
                        11080
                        11100
                        11120
                        11140
                        11160
                        11180
                        11200
                        11220
                        11240
                        11260
                        11280
                        11300
                        11320
                        11340
                        11360
```

```
                                GETSPA RETURNS THE LOCATION OF THE AVAILABLE
                                SPACE.
                                4) CREATE THE NEW VALUE BY COPYING PARTS
                                OF THE ARGUMENTS OR WHATEVER.
                                5) FREE UP THE ARGUMENTS BY CALLING FRETMP.
                                6) JUMP TO PUTNEW TO GET THE DESCRIPTOR IN
                                DSCTMP TRANSFERRED INTO A NEW STRING TEMPORARY.

                        THE REASON FOR STRING TEMPORARIES IS THAT GARBAGE
                        COLLECTION HAS TO KNOW ABOUT ALL ACTIVE STRING DESCRIPTORS
                        SO IT KNOWS WHAT IS AND ISN'T IN USE. STRING TEMPORARIES ARE
                        USED TO STORE THE DESCRIPTORS OF STRING EXPRESSIONS.

                        INSTEAD OF HAVING AN ACTUAL VALUE STORED IN THE
                        FAC, AND HAVING THE VALUE OF A TEMPORARY RESULT
                        BEING SAVED ON THE STACK, AS HAPPENS WITH NUMERIC
                        VARIABLES, STRINGS HAVE THE POINTER TO A STRING DESCRIPTOR
                        STORED IN THE FAC, AND IT IS THIS POINTER
                        THAT GETS SAVED ON THE STACK BY FORMULA EVALUATION.
                        STRING FUNCTIONS CANNOT FREE THEIR ARGUMENTS UP RIGHT
                        AWAY SINCE GETSPA MAY FORCE
                        GARBAGE COLLECTION AND THE ARGUMENT STRINGS
                        MAY BE OVER-WRITTEN SINCE GARBAGE COLLECTION
                        WILL NOT BE ABLE TO FIND AN ACTIVE POINTER TO
                        THEM. FUNCTION AND OPERATOR RESULTS ARE BUILT IN
                        DSCTMP SINCE STRING TEMPORARIES ARE ALLOCATED
                        (PUTNEW) AND DEALLOCATED (FRETMP) IN A FIFO ORDERING
                        (I.E., A STACK) SO THE NEW TEMPORARY CANNOT
                        BE SET UP UNTIL THE OLD ONE(S) ARE FREED, TRYING
                        TO BUILD A RESULT IN A TEMPORARY AFTER
                        FREEING UP THE ARGUMENT TEMPORARIES COULD RESULT
                        IN ONE OF THE ARGUMENT TEMPORARIES BEING OVERWRITTEN
                        TOO SOON BY THE NEW RESULT.

                        STRING SPACE IS ALLOCATED AT THE VERY TOP
                        OF MEMORY, MEMSIZ POINTS BEYOND THE LAST LOCATION OF
                        STRING SPACE, STRING ARE STORED IN HIGH LOCATIONS
                        FIRST. WHENEVER STRING SPACE IS ALLOCATED (GETSPA)
                        FRETOP, WHICH IS INITIALIZED TO (MEMSIZ), IS UPDATED
                        TO GIVE THE HIGHEST LOCATION IN STRING SPACE
                        THAT IS NOT IN USE, THE RESULT IS THAT
                        FRETOP GETS SMALLER AND SMALLER, UNTIL SOME
                        ALLOCATION WOULD MAKE (FRETOP) LESS THAN OR EQUAL TO
                        (STKTOP). THIS MEANS STRING SPACE HAS RUN INTO THE
                        STACK AND THAT GARBAGE COLLECTION MUST BE CALLED.

                        GARBAGE COLLECTION:
                                0. MINPTR=(STKTOP) (FRETOP)=(MEMSIZ)
                                1. REMMIN=0
                                2. FOR EACH STRING DESCRIPTOR
                                (TEMPORARIES, SIMPLE STRINGS, STRING ARRAYS)
                                IF THE STRING IS NOT NULL AND ITS POINTER IS
                                .GT.MINPTR AND .LT.FRETOP,
```

```
                                        MINPTR=THIS STRING DESCRIPTORS POINTER
                                        REMMIN=POINTER AT THIS STRING DESCRIPTOR
                                        END
                                        3, IF REMMIN,NE,0 (WE FOUND AN UNCOLLECTED STRING)
                                        BLOCK TRANSFER THE STRING DATA POINTED
                                        TO IN THE STRING DESCRIPTOR POINTED TO BY REMMIN
                                        SO THAT THE LAST BYTE OF STRING DATA IS AT
                                        (FRETOP), UPDATE FRETOP SO THAT IT
                                        POINTS TO THE LOCATION JUST BELOW THE ONE
                                        THE STRING DATA WAS MOVED INTO, UPDATE
                                        THE POINTER IN THE DESCRIPTOR SO IT POINTS
                                        TO THE NEW LOCATION OF THE STRING DATA,
                                        GO TO STEP 1,

                                AFTER CALLING GARBAGE COLLECTION GETSPA AGAIN CHECKS
                                TO SEE IF [A] CHARACTERS ARE AVAILABLE BETWEEN
                                (STKTOP) AND (FRETOP) , IF NOT AN "OUT OF STRING"
                                ERROR IS INVOKED,

                        MATH PACKAGE
                                THE MATH PACKAGE CONTAINS FLOATING INPUT (FIN),
                                FLOATING OUTPUT (FOUT) FLOATING COMPARE (FCOMP)
                                ,,, AND ALL THE NUMERIC OPERATORS AND FUNCTIONS,
                                THE FORMATS,CONVENTIONS AND ENTRY POINTS ARE ALL
                                DESCRIBED IN THE MATH PACKAGE ITSELF,

                        INIT == THE INITIALIZATION ROUTINE
                                INITIALIZATION FIRST LOOKS AT THE SWITCH REGISTER
                                TO SEE WHAT TYPE OF I/O SHOULD BE DONE,
                                ANY NON-STANDARD I/O CAUSES LOCATIONS IN BASIC
                                TO BE CHANGED, THEN THE AMOUNT OF MEMORY,
                                TERMINAL WIDTH,AND WHICH FUNCTIONS TO BE RETAINED
                                ARE ASCERTAINED FROM THE USER, A ZERO IS PUT DOWN
                                AT THE FIRST LOCATION NOT USED BY THE MATH-PACKAGE
                                AND TXTTAB IS SET UP TO POINT AT THE NEXT LOCATION,
                                THIS DETERMINES WHERE PROGRAM STORAGE WILL START, THE
                                HIGHEST MEMORY LOCATION MINUS THE AMOUNT OF DEFAULTED
                                STRING SPACE (50) GIVES THE FIRST LOCATION USED BY THE
                                STACK, SPECIAL CHECKS ARE MADE TO MAKE SURE
                                ALL QUESTIONS IN INIT ARE ANSWERED REASONABLY, SINCE
                                ONCE INIT FINISHES THE LOCATIONS IT USES ARE
                                USED FOR PROGRAM STORAGE, THE LAST THING INIT DOES IS
                                CHANGE LOCATION ZERO TO BE A JUMP TO READY INSTEAD
                                OF INIT, ONCE THIS IS DONE THERE IS NO WAY TO RESTART
                                INIT,

                        STORAGE
                                                A ZERO,
                                [TXTTAB]        POINTER TO NEXT LINE'S POINTER
                                                LINE # OF THIS LINE (2 BYTES)
                                                CHARACTERS ON THIS LINE
                                                ZERO
                                                POINTER AT NEXT LINE'S POINTER
```



```
                                                        (POINTED TO BY THE ABOVE POINTER)
                                                ,,, REPEATS ,,,
                                LAST LINE:      POINTER AT ZERO POINTER
                                                LINE # OF THIS LINE
                                                CHARACTERS ON THIS LINE
                                                ZERO
                                                DOUBLE ZERO (POINTED TO BY THE ABOVE POINTER)
                                [VARTAB]        SIMPLE VARIABLES, 6 BYTES PER VALUE,
                                                2 BYTES GIVE THE NAME, 4 BYTES THE VALUE
                                                ,,, REPEATS ,,,
                                [ARYTAB]        ARRAY VARIABLES, 2 BYTES NAME, 2 BYTE
                                                LENGTH, VALUE (EXTRA IF MULDIM ON)
                                                ,,, REPEATS ,,,
                                [STKEND]        FREE SPACE
                                                ,,, REPEATS ,,,
                                                MOST RECENT STACK ENTRY
                                                ,,, REPEATS ,,,
                                [STKTOP]        FIRST STACK ENTRY
                                                FREE STRING SPACE
                                                ,,, REPEATS ,,,
                                [FRETOP]        STRING SPACE IN USE
                                                ,,, REPEATS ,,,
                                [MEMSIZ]        HIGHEST MACHINE LOCATION
                                                UNUSED EXCEPT BY THE VAL FUNCTION,

                        HIGH LOCATIONS

                        *
                        PAGE
```

```
                                13000  SUBTTL  RST ROUTINES
                                13020  RELOC 0
000000  000303  000000  000000  13040  START:  DI                      ;DISENABLE INTERRUPTS
000001  001000  000303          13060          JMP     INIT            ;INIT IS THE INITIALIZE ROUTINE
000002  000000  000000*
000003  000000  000000
                                13080                                  ;IT SETS UP CERTAIN
                                13100                                  ;LOCATIONS DELETES FUNCTIONS IF
                                13120                                  ;DESIRED AND
                                13140                                  ;CHANGES THIS TO JMP READY
                                13160  IFN     LENGTH-2,<
                                13180          ADR(DEINT)              ;STORE HERE THE ROUTINE
                                13200                                  ;TO TURN THE FAC INTO
                                13220                                  ;A TWO-BYTE SIGNED INTEGER
                                13240          ADR(GIVABF)>            ;STORE HERE THE ADDRESS
                                13260                                  ;OF THE ROUTINE TO CONVERT [A,B]
                                13280                                  ;TO A FLOATING POINT NUMBER IN THE FAC
                                13300  IFE     LENGTH-2,<
000004  000000  000000*         13320          ADR(FXCINT)             ;TURN FAC INTO AN INTEGER IN [H,L]
000005  000000  000002
000006  000002  000000*         13340          ADR(MAKINT)>            ;TURN [H,L] INTO A VALUE IN THE FAC
000007  000000  000004
                                13360                                  ;SET VALTYP FOR INTEGER
000010                          13380  RELOC 8
                                13400  ;
                                13420  ; SYNCHK LOOKS AT THE CURRENT CHARACTER TO MAKE SURE IT
                                13440  ; IS A SPECIFIC THING (CONTAINED IN THE LOCATION AFTER THE CALL)
                                13460  ; IF NOT IT CALLS THE 'SYNTAX ERROR' ROUTINE, OTHERWISE IT GOBBLES
                                13480  ; THE NEXT CHARACTER AND RETURNS. (BY FALLING INTO CHRGET)
                                13500  ;
                                13520  ; ALL REGISTERS ARE PRESERVED EXCEPT [A]=NEW CHAR
                                13540  ; AND [H,L] ENDS UP POINTING AT THE CHARACTER AFTER THE ONE
                                13560  ; WHICH WAS CHECKED.
                                13580  ;
000010  001000  000176          13600          MOV     A,M             ;GET THE CURRENT CHARACTER
000011  001000  000343          13620          XTHL                    ;GET CALL ADDRESS INTO [H,L]
                                13640                                  ;PUT TEXT POINTER ON STACK
000012  001000  000276          13660          CMP     M               ;SEE IF [A] =CURRENT CHARACTER
                                13680                                  ;IS THE RIGHT THING,
000015  001000  000043          13700          INX     H               ;FIX RETURN ADDRESS,
000014  001000  000343          13720          XTHL                    ;PUT RETURN ADDRESS BACK AND RESTORE
                                13740                                  ;THE TEXT POINTER,
000015  001000  000302          13760          JNZ     SNERR           ;IF THE CHARACTER WASN'T RIGHT CALL
000016  000000  002076'
000017  000200  000006'
                                13780                                  ;THE 'SYNTAX ERROR' ROUTINE,
                                13800                                  ;OTHERWISE FALL THROUGH
                                13820                                  ;AND GET ANOTHER CHARACTER.
                                13840  ;
                                13860  ; CHRGET, USING [H,L] AS THE CURRENT TEXT POINTER FETCHES
                                13880  ; A NEW CHARACTER INTO [A] AFTER INCREMENTING [H,L]
                                13900  ; AND SETS CONDITION CODES ACCORDING TO WHATS IN [A]
                                13920  ;       C=        NUMERIC     ("0" THROUGH "9")
```



```
                                13940  ;       Z=        ":" OR END-OF-LINE (A 0)
                                13960  ;
                                13980  ; ALL REGISTERS SAVED EXCEPT [A]=NEW CHAR
                                14000  ; [H,L]=[H,L]+1
                                14020  ;
020  000020                     14040  RELOC 16
                                14060  IFE     LENGTH,<CHRGTH1>
000020  001000  000043          14080          INX     H               ;UPDATE THE TEXT POINTER
000021  001000  000176          14100          MOV     A,M             ;GET NEW CHARACTER
000022  001000  000376          14120          CPI     ":"             ;MAKE ":" HAVE ZERO ON AND
000023  000000  000072
                                14140                                  ;CARRY OFF
                                14160                                  ;ALL ALPHABETICS & RESERVED
                                14180                                  ;WORDS GET ZERO & CARRY OFF
000024  001000  000320          14200          RNC                     ;,GT.":" GO BACK
000025  001000  000303          14220          JMP     CHRCON          ;NO ROOM FOR WHOLE ROUTINE
000026  000000  003433'
000027  000000  000016'
                                14240  ;
                                14260  ; THIS RST ROUTINE OUTPUTS THE CHARACTER IN [A] USING PRTFLG (LPT OR TTY)
                                14280  ; CNTRFL (SUPRESS OUTPUT OR NOT), TTYPOS (PRINT HEAD POSITION),
                                14300  ; TIMING ETC,, NO REGISTERS OR CONDITION CODES ARE CHANGED.
                                14320  ;
030  000030                     14340  RELOC 24
000030  001000  000365          14360  OUTDO:  PUSH    PSW
                                14380  IFN     CONTRH,<
000031  001000  000272          14400          LDA     CNTRFL          ;GET SUPPRESS FLAG
000032  000000  001541'
000033  000000  000026'
000034  001000  000267          14420          ORA     A>              ;SEE IF IT IS SET
                                14440  IFE     LENGTH!CONTRH!LPTSW,<
                                14460          LDA     TTYPOS>         ;USE RST BYTES, [A]=TTYPOS
000035  001000  000305          14480          JMP     OUTCON
000036  000000  003665'
000037  000000  000032'
                                14500  ;
                                14520  ; COMPAR COMPARES [H,L] WITH [D,E] UNSIGNED
                                14540  ;
                                14560  ;               [H,L] LESS THAN [D,E] SET CARRY
                                14580  ;               [H,L] = [D,E] SET ZERO
                                14600  ;
                                14620  ; [A] IS THE ONLY REGISTER USED
                                14640  ;
040  000040                     14660  RELOC 32
000040  001000  000174          14680          MOV     A,H
000041  001000  000222          14700          SUB     D
000042  001000  000300          14720          RNZ
000043  001000  000175          14740          MOV     A,L
000044  001000  000223          14760          SUB     E
000045  001000  000311          14780          RET
                                14800
000046  000000  000001          14820  NULCNT: 1                       ;STORE HERE THE NUMBER OF NULLS
                                14840                                  ;TO PRINT AFTER CRLF
```

```
                                14860   TTYPOS: BLOCK   1               ;STORE TERMINAL POSITION HERE
                                14880   ;
                                14900   ;THE FSIGN  RST RETURNS A=+1 IF FAC IS LESS THAN 0
                                14920   ;       A=0 IF FAC=0
                                14940   ;       A=1 IF FAC GREATER THAN ZERO
                                14960   ; THE CONDITION CODES REFLECT THE VALUE OF (A)
                                14980   ; AND NO OTHER REGISTERS ARE MODIFIED.
                                15000   ; THIS WORKS ONLY WHEN THE FAC IS A SINGLE OR DOUBLE PRECISION NUMBER
                                15020   ; THE 'FSIGN' ROUTINE IS MORE GENERAL SINCE
                                15040   ; IT WILL TAKE THE SIGN OF INTEGERS AS WELL
                                15050   ; AND GIVES "TMERR" ON STRINGS.
                                15060   ;
000050'                         15080   RELOC   40
000050' 001000  000072          15100   SIGN:   LDA     FAC
000051' 000000  001642'
000052' 000000  000036'
000053' 001000  000267          15120           ORA     A
000054' 001000  000302          15140           JNZ     SIGNC
000055' 000000  000000*
000056' 000000  000051'
000057' 001000  000311          15160           RET
                                15180   ;
                                15200   ; THIS IS THE PUSHM RST
                                15220   ; EFFECT IS:
                                15240   ;       MOV     C,M
                                15260   ;       INX     H
                                15280   ;       MOV     B,M
                                15300   ;       INX     H
                                15320   ;       PUSH    B
                                15340   ; DIFFICULTY COMES IN BECAUSE OF THE
                                15360   ; RETURN ADDRESS.
                                15380   ;
060060'                         15400   RELOC   48
060060' 001000  000343          15420           XTHL                    ;SWITCH [H,L] AND RETURN ADDRESS
060061' 001000  000042          15440           SHLD    PUSHMA+1        ;FIXUP JUMP TO PLACE TO GO
060062' 000000  000101'
060063' 000000  000055'
060064' 001000  000341          15460           POP     H               ;REGAIN [H,L]
                                15480   IFN     LENGTH,<
060065' 001000  000303          15500           JMP     $CODE+59        ;IN 8K ALLOW USER TO HAVE RST 7
060066' 000000  000073'
060067' 000000  000062'
                                15520                                   ;FOR INTERRUPT TRAPPING
000070'                         15540   RELOC   56
000070' 001000  000311          15560           RET                     ;INITIALLY NO INTERRUPT
                                15580                                   ;ROUTINE
000071' 001000  000000          15600           NOP
000072' 001000  000000          15620           NOP>
000073' 001000  000116          15640           MOV     C,M             ;GRAB FROM MEMORY
000074' 001000  000043          15660           INX     H
000075' 001000  000106          15680           MOV     B,M
000076' 001000  000043          15700           INX     H
000077' 001000  000325          15720           PUSH    B               ;PUSH [B,C] ONTO THE STACK
```



```
                                15740                                   ;SINCE IT CONTAINS (M)
000100' 001000  000303          15760   PUSHMA: JMP     PUSHMA          ;RETURN  ADDRESS STORED HERE
000101' 000000  000100'
000102' 000000  000066'
                                15800   PAGE
```

```
                                        15820  SUBTTL DISPATCH TABLES,RESERVED WORDS, ERROR TEXT... ALL CONSTANT
                                        15840
000103' 000000  000000*                 15860  FUNDSP: ADR(SGN)
000104' 000000  000000'                 15880          ADR(INT)>
                                        15900          IFN     LENGTH=2,<
                                        15920          IFE     LENGTH=2,<
000105' 000000  000000*                 15940          ADR(VINT)>
000106' 000000  000103'                 15960          ADR(ABS)
000110' 000000  000105'                 15980  USRLOC: ALM(ILLFUN)              ;INITIALLY NO USER ROUTINE
000111' 000000  010776'                 16000  IFN     LENGTH,<ADR(FRE)
000113' 000000  007337'                 16020          ADR(FNINP)
000114' 000000  000111'                 16040  IFN     LPTSW,<ADR(LPOS)>
000115' 000000  010712'                 16060          ADR(POS)>
000116' 000000  000113'                 16080  SGRFIX: ADR(SGR)
                                        16100  RNDFIX: ADR(RND)
000117' 000000  007406'                 16120  IFN     EXTFNC,<
000120' 000000  000115'                 16140          ADR(LOG)
000121' 000000  000117'                 16160          ADR(EXP)
000123' 000000  000000*                 16180  COSFIX: ADR(COS)>
000125' 000000  000120'                 16200  SINFIX: ADR(SIN)
000127' 000000  000123'                 16220  IFN     EXTFNC,<
000130' 000000  000000*                 16240  TANFIX: ADR(TAN)
000132' 000000  000125'                 16260  ATNFIX: ADR(ATN)>
000134' 000000  000127'                 16280  IFN     LENGTH,<
000135' 000000  011314'                 16300          ADR(PEEK)>
000137' 000000  000132'                 16320  IFN     DSKFUN,<ADR(DSK1$)>
                                        16340  IFN     STRING,<
000140' 000000  010541'                 16360          ADR(LEN)
000141' 000000  007564'                 16380          ALM(STR$)
000143' 000000  011042'                 16400          ADR(VAL)
000145' 000000  010515'                 16420          ADR(ASC)
000147' 000000  010532'                 16440          ADR(CHR$)
000151' 000000  000151'
```



```
000153' 000000  010552'                 16460          ADR(LEFT$)
000155' 000000  000153'
000157' 000000  010531'                 16480          ADR(RIGHT$)
000160' 000000  000155'
000161' 000000  010543'                 16500          ADR(MID$)>
000162' 000000  000157'

                                        16540  DEFINE ADRP(X),<ADR(X)>
                                        16560  IFE     LENGTH=2,<
                                        16580  DEFINE ADRP(X),<>>
000163' 000000  000171                  16600  OPTAB:  121                      ;OPERATOR TABLE CONTAINS
                                        16620                                   ;PRECEDENCE FOLLOWED BY
                                        16640                                   ;THE ROUTINE ADDRESS
                                        16660          ADRP(PADDT)
000164' 000000  000171                  16680          121
                                        16700          ADRP(PSUBT)
000165' 000000  000173                  16720          123
                                        16740          ADRP(PMULT)
000166' 000000  000173                  16760          123
                                        16780          ADRP(PDIVT)
000167' 000000  000177                  16800  IFN     EXTFNC,<127
                                        16820          ADRP(PPWRT)>
                                        16840  IFN     LENGTH,<
000170' 000000  000120                  16860          80
                                        16880          ADRP(AN.)
000171' 000000  000106                  16900          70
                                        16920          ADRP(LR)>

                                        16960  ;
                                        16980  ; TOKENS FOR RESERVED WORDS ALWAYS HAVE THE MOST
                                        17000  ; SIGNIFIGANT BIT ON
                                        17020  ; THE LIST OF RESERVED WORDS
                                        17040  ;
                        000177          17060  J=128-1
                                        17080  DEFINE  DCI(A),<DB^J+1
                                        17090          XLIST
                                        17100          DC(A)
                                        17110          LIST>
                        000200          17140  ENDTK==J
                        000201          17160  FORTK==J
                        000203          17240  DATATK==J
                        000210          17260  GOTOTK==J
                        000212          17420  IFTK==J
                        000214          17480  GOSUTK=J
                        000216          17540  REMTK=J
                                        17660  IFE     LENGTH=2,<
                        000220          17700          ELSETK==J
                                        17720  IFN     DSKFUN,<DCI"DSK!$">
                                        17800  IFN     LPTSW,<DCI"LPRINT">
                                        17820  IFN     LENGTH,<
                        000231          17840          PRINTK==J
                                        17940  IFE     REALIO,<
                                        17960          DCI"OUT">
```

```
                                                18000   IFN     LPTSW,<UCI"LLIST>
                                                18060   IFN     CASSW,<UCI"CLOAD"
                                                18080           UCI"CSAVE">
                                                18100   IFN     CONSSW,<UCI"CONSOLE">
                        000237                  18140           SCMATK==Q
                                                18160   ; END OF COMMAND LIST
1007    000371' 000000  000120                  18180           "T"
1008    000372' 000000  000101                  18200           "A"
1009    000373' 000000  000102                  18220           "B"
1010    000374' 000000  000058                  18240           "("+128
1011                    000240                  18260           Q==Q+1
1012                    000240                  18280           TABTK==Q
1013                    000241                  18320           TOTK==Q
1014                                            18340   IFN     LENGTH,<
1015    000377' 000000  000123                  18360           "S"
1016    000400' 000000  000122                  18380           "R"
1017    000401' 000000  000103                  18400           "C"
1018    000402' 000000  000050                  18420           "("+128                 ;MACRO DOESNT LIKE ("S IN ARGUMENTS
1019                    000242                  18440           Q==Q+1
1020                    000242                  18460           SPCTK==Q>
1021                    000243                  18520           FNTK==Q>
1022                    000244                  18540           USINTK==Q>
1023                    000245                  18580           THENTK==Q
1024                                            18600   IFN     LENGTH,<
1025                    000246                  18640           NOTTK==Q>
1026                    000247                  18660           STEPTK==Q
1027                    000250                  18720           PLUSTK==Q
1028                    000251                  18760           MINUTK==Q
1029                    000007                  18800           LSTOPK==Q+1-PLUSTK      ;CRUNCH # OF HIGHEST OP+1=PLUSTK
1030    000437' 000000  000270                  18900           190                     ;A GREATER THAN SIGN
1031                    000257                  18920           Q==Q+1
1032                    000257                  18940           GREATK==Q
1033                    000260                  18980           EQULTK==Q
1034    000441' 000000  000274                  19000           188
1035                    000261                  19020           Q==Q+1                  ;A LESS THAN SIGN
1036                    000261                  19040           LESSTK==Q
1037                                            19060   ;
1038                                            19080   ; NOTE DANGER OF ONE RESERVED WORD BEING A PART
1039                                            19100   ; OF ANOTHER
1040                                            19120   ; IE , , IF 2 GREATER THAN F OR T=S THEN...
1041                                            19140   ; WILL NOT WORK,II SINCE "FOR" WILL BE CRUNCHED!!
1042                                            19160   ; IN ANY CASE MAKE SURE THE SMALLER WORD APPEARS
1043                                            19180   ; SECOND IN THE RESERVED WORD TABLE ("INP" AND "INPUT")
1044                                            19190   ; ANOTHER EXAMPLE: IF T OR Q THEN ... "TO" IS CRUNCHED
1045                                            19200   ;
1046                    000262                  19260           QNEFUN==Q
1047                                            19360   IFN     LPTSW,<DCI"_POS">
1048                    000271                  19420           SQRTK==Q
1049                                            19460   IFN     EXTFNC,<
1050                    000100                  19600           ATNTK==Q>
1051                                            19620   IFN     LENGTH,<
1052                                            19660   IFN     OSKFUN,<DCI"OSKIS">
1053                                            19680   IFN     STRING,<
```



```
                        000306                  19800   LASNUM==Q                       ;NUMBER OF LAST FUNCTION
                                                19820                                   ;THAT TAKES ONE ARG
000565' 000000  000000                          19900           0                       ;MARKS END OF RESERVED WORD LIST
                                                19940   STMDSP: ADR(END)
000566' 000000  003474'                         19960           ADR(FOR)
000567' 000000  000161'                         19980           ADR(NEXT)
000570' 000000  000564'                         20000           ADR(DATA)
000572' 000000  000572'                         20020           ADR(INPUT)
000574' 000000  000572'                         20040           ADR(DIM)
000576' 000000  006500'                         20060           ADR(READ)
000600' 000000  004760'                         20080           ADR(LET)
000602' 000000  006610'                         20100           ADR(GOTO)
000604' 000000  004010'                         20120           ADR(RUN)
000606' 000000  003754'                         20140           ADR(IF)
000610' 000000  002606'                         20160           ADR(RESTORE)
000612' 000000  002616'                         20180           ADR(GOSUB)
000614' 000000  003770'                         20200           ADR(RETURN)
000616' 000000  004044'                         20220           ADR(REM)
000620' 000000  004074'                         20240           ADR(STOP)
000622' 000000  003472'                         20260   IFE     LENGTH=2,<
000624' 000000  004074'                         20280           ADR(ELSE)
000626' 000000  006622'                         20300           ADR(TON)
000630' 000000  003604'                         20320           ADR(TOFF)
000632' 000000  000626'                         20340           ADR(EDIT)>
000634' 000000  010725'                         20360   IFN     LENGTH,<ADR(FNOUT)
000636' 000000  004271'                         20380           ADR(ONGOTO)
000637' 000000  000634'                         20400           ADR(NULL)
000640' 000000  003566'                         20420           ADR(FNWAIT)>
000641' 000000  000636'
000642' 000000  010733'
000643' 000000  000640'
```

```
BASIC MOS B080  GATES/ALLEN/DAVIDOFF     MACRO 47(113) 03:12 10-SEP-75 PAGE 4-4
F3      MAC      6-SEP-64 03:11          DISPATCH TABLES,RESERVED WORDS, ERROR TEXT... ALL CONSTANT

                                         20440   IFN     DSKFUN,<ADR(DSKU$)>
                                         20460   IFN     LPTSW,<ADR(LPRINT)>
                                         20480   IFN     LENGTH,<
000644' 000000  011323'                 20500           ADR(POKE)>
000645' 000000  000642'
000646' 000000  004411'                  20520           ADR(PRINT)
000647' 000000  014044'
000650' 000000  007411'                  20540   IFN     FUNCTS,<ADR(DEF)>
000651' 000000  000646'
                                         20580   IFN     LENGTH,<ADR(CONT)>
000652' 000000  003542'
000653' 000000  000650'
                                         20600   IFE     REALIO,<ADR(DDT)>
000654' 000000  011672'                  20620           ADR(LIST)
000655' 000000  000652'
                                         20640   IFN     LPTSW,<ADR(LLIST)>
000656' 000000  011245'                  20660   IFE     LENGTH-2,<ADR(DELETE)>
000657' 000000  000654'
000660' 000000  003703'                  20680           ADR(CLEAR)
000661' 000000  000656'
                                         20700   IFN     LASSW,<ADR(LLOAD)
                                         20720           ADR(CSAVE)>
                                         20740   IFN     CONSSW,<ADR(CONSOLE)>
000662' 000000  002421'                  20760           ADR(SCRATH)
000663' 000000  000660'

                                         20800   IFE     LENGTH-2,<
000664' 000000  000000*          FRCTBL: ADR(FRCDBL)
000665' 000000  000662'
000666' 000000  000000*                  20840           ADR(FNCINT)
000667' 000000  000664'
000670' 000000  000000*                  20860           ADR(FNCSNG)
000671' 000000  000666'

                                         20880   ;
                                         20900   ; THESE TABLES ARE USED AFTER THE DECISION HAS BEEN MADE
                                         20920   ; TO APPLY AN OPERATOR AND ALL THE NECESSARY CONVERSION HAS
                                         20940   ; BEEN DONE TO MATCH THE TWO ARGUMENT TYPES (APPLOP)
                                         20960   ;
000672' 000000  000000*          DBLDSP: ADR(DADD)               ;DOUBLE PRECISION ROUTINES
000673' 000000  000670'
000674' 000000  000000*                  21000           ADR(DSUB)
000675' 000000  000672'
000676' 000000  000000*                  21020           ADR(DMULT)
000677' 000000  000674'
000700' 000000  000000*                  21040           ADR(DDIV)
000701' 000000  000676'
000702' 000000  000000*          21060   ADR(DCOMP)
000703' 000000  000700'
000704' 000000  000000*          21080   SNGDSP: ADR(FADD)               ;SINGLE PRECISION ROUTINES
000705' 000000  000702'
000706' 000000  000000*          21100           ADR(FSUB)
000707' 000000  000704'
000710' 000000  000000*          21120           ADR(FMULT)
```

```
BASIC MOS B080  GATES/ALLEN/DAVIDOFF     MACRO 47(113) 03:12 10-SEP-75 PAGE 4-5
F3      MAC      6-SEP-64 03:11          DISPATCH TABLES,RESERVED WORDS, ERROR TEXT... ALL CONSTANT

000711' 000000  000706'
000712' 000000  000000*                  21140           ADR(FDIV)
000713' 000000  000710'
000714' 000000  000000*          21160   ADR(FCOMP)
000715' 000000  000712'
000716' 000000  000000*          21180   INTDSP: ADR(IADD)               ;INTEGER ROUTINES
000717' 000000  000714'
000720' 000000  000000*                  21200           ADR(ISUB)
000721' 000000  000716'
000722' 000000  000000*                  21220           ADR(IMULT)
000723' 000000  000720'
000724' 000000  000000*                  21240           ADR(IDIV)
000725' 000000  000722'
000726' 000000  000000*                  21260           ADR(ICOMP)>
000727' 000000  000724'

                                         21300
                 777777  777776         21320   Q=-2
                                         21340   DEFINE  DCL(X),<X>
                                         21360   DEFINE  DCE(X),<Q=Q+2
                                         21370           XLIST
                                         21380           DC(X)
                                         21390           LIST>

000730'                                  21420   ERRTAB:
                                         21440   IFE     LENGTH-2,<
000730' 000000  000000           21460           0
                                 21480           Q=0
                                         21500           DEFINE  DCE(X),<>
                                         21520           DEFINE  DCL(X),<
                                         21540           Q=Q+1
                                         21560           DC(X)
                                         21580           0>>
                                         21600           DCE"NF"
000731' 000000  000110           21620           DCL"NEXT WITHOUT FOR"
000732' 000000  000105
000733' 000000  000130
000734' 000000  000124
000735' 000000  000040
000736' 000000  000127
000737' 000000  000111
000740' 000000  000124
000741' 000000  000110
000742' 000000  000117
000743' 000000  000125
000744' 000000  000124
000745' 000000  000040
000746' 000000  000105
000747' 000000  000117
000750' 000000  000122
000751' 000000  000000
                 000001          21640           ERRNF==Q
```

```
1213                                            21660           DCE"SN"
1214  000752' 000000  000123                    21680           DCL"SYNTAX ERROR"
1215  000753' 000000  000131
1216  000754' 000000  000116
1217  000755' 000000  000124
1218  000756' 000000  000101
1219  000757' 000000  000130
1220  000760' 000000  000040
1221  000761' 000000  000005
1222  000762' 000000  000022
1223  000763' 000000  000126
1224  000764' 000000  000017
1225  000765' 000000  000122
1226  000765' 000000  000040
1227  000766' 000000  000000
1228                  000002            21700           ERRSN=.-2
1229                                    21720           DCE"RG"
1230  000767' 000000  000122            21740           DCL"RETURN WITHOUT GOSUB"
1231  000770' 000000  000005
1232  000771' 000000  000024
1233  000772' 000000  000025
1234  000773' 000000  000122
1235  000774' 000000  000116
1236  000775' 000000  000040
1237  000776' 000000  000127
1238  000777' 000000  000111
1239  001000' 000000  000124
1240  001001' 000000  000010
1241  001002' 000000  000117
1242  001003' 000000  000025
1243  001004' 000000  000124
1244  001005' 000000  000040
1245  001006' 000000  000107
1246  001007' 000000  000117
1247  001010' 000000  000125
1248  001011' 000000  000105
1249  001012' 000000  000040
1250  001013' 000000  000000
1251  001015' 000000  000003            21760           ERRNG=.-2
1252                                    21780           DCE"DL"
1253  001014' 000000  000117            21800           DCL"OUT OF DATA"
1254  001015' 000000  000125
1255  001016' 000000  000124
1256  001017' 000000  000040
1257  001020' 000000  000117
1258  001021' 000000  000106
1259  001022' 000000  000040
1260  001023' 000000  000104
1261  001024' 000000  000101
1262  001025' 000000  000124
1263  001026' 000000  000101
1264  001027' 000000  000301
```



```
1266  001027' 000000  000000
1267                  000004            21820           ERRDD=.-2
1268                                    21840           DCE"FC"
1269  001030' 000000  000111            21860           DCL"ILLEGAL FUNCTION CALL"
1270  001031' 000000  000114
1271  001032' 000000  000114
1272  001033' 000000  000105
1273  001034' 000000  000107
1274  001035' 000000  000101
1275  001036' 000000  000114
1276  001037' 000000  000040
1277  001040' 000000  000105
1278  001041' 000000  000125
1279  001042' 000000  000116
1280  001043' 000000  000103
1281  001044' 000000  000124
1282  001045' 000000  000111
1283  001046' 000000  000117
1284  001047' 000000  000116
1285  001050' 000000  000040
1286  001051' 000000  000103
1287  001052' 000000  000101
1288  001053' 000000  000114
1289  001054' 000000  000114
1290  001054' 000000  000314
1291  001055' 000000  000000
1292                  000005            21880           ERRFC=.-2
1293                                    21900           DCE"OV"
1294  001056' 000000  000117            21920           DCL"OVERFLOW"
1295  001057' 000000  000126
1296  001060' 000000  000105
1297  001061' 000000  000122
1298  001062' 000000  000106
1299  001063' 000000  000114
1300  001064' 000000  000117
1301  001065' 000000  000127
1302  001065' 000000  000327
1303  001066' 000000  000000
1304                  000006            21940           ERROV=.-2
1305                                    21960           DCE"OM"
1306  001067' 000000  000117            21980           DCL"OUT OF MEMORY"
1307  001070' 000000  000125
1308  001071' 000000  000124
1309  001072' 000000  000040
1310  001073' 000000  000117
1311  001074' 000000  000106
1312  001075' 000000  000040
1313  001076' 000000  000115
1314  001077' 000000  000105
1315  001100' 000000  000115
1316  001101' 000000  000117
1317  001102' 000000  000124
1318  001103' 000000  000131
```

```
1319  001103' 000000  000331
1320  001104' 000000  000000
1321                  000007          22000           ERRQM==.
1322                                  22020           DCF".S"
1323  001105' 000000  000125          22040           DCL"?UNDEFINED STATEMENT"
1324  001106' 000000  000116
1325  001107' 000000  000.04
1326  001110' 000000  000.05
1327  001111' 000000  000106
1328  001112' 000000  000111
1329  001113' 000000  000116
1330  001114' 000000  000.03
1331  001115' 000000  000104
1332  001116' 000000  000040
1333  001117' 000000  000123
1334  001120' 000000  000124
1335  001121' 000000  000101
1336  001122' 000000  000124
1337  001123' 000000  000105
1338  001124' 000000  000115
1339  001125' 000000  000.05
1340  001126' 000000  000116
1341  001127' 000000  000124
1342  001127' 000000  000324
1343  001130' 000000  000000
1344                  000010          22060           ERRUS==.
1345                                  22080           DCF"BS"
1346  001131' 000000  000123          22100           DCL"SUBSCRIPT OUT OF RANGE"
1347  001132' 000000  000125
1348  001133' 000000  000102
1349  001134' 000000  000123
1350  001135' 000000  000103
1351  001136' 000000  000122
1352  001137' 000000  000111
1353  001140' 000000  000120
1354  001141' 000000  000124
1355  001142' 000000  000040
1356  001143' 000000  000117
1357  001144' 000000  000125
1358  001145' 000000  000124
1359  001146' 000000  000040
1360  001147' 000000  000117
1361  001150' 000000  000106
1362  001151' 000000  000040
1363  001152' 000000  000122
1364  001153' 000000  000101
1365  001154' 000000  000116
1366  001155' 000000  000107
1367  001156' 000000  000105
1368  001156' 000000  000325
1369  001157' 000000  000000
1370                  000011          22120           ERRBS==.
1371                                  22140           DCF"DV"
```



```
1372  001160' 000000  000122          22160           DCL"REDIMENSIONED ARRAY"
1373  001161' 000000  000105
1374  001162' 000000  000104
1375  001163' 000000  000111
1376  001164' 000000  000115
1377  001165' 000000  000105
1378  001166' 000000  000116
1379  001167' 000000  000123
1380  001170' 000000  000111
1381  001171' 000000  000117
1382  001172' 000000  000116
1383  001173' 000000  000105
1384  001174' 000000  000104
1385  001175' 000000  000040
1386  001176' 000000  000101
1387  001177' 000000  000122
1388  001200' 000000  000122
1389  001201' 000000  000101
1390  001202' 000000  000131
1391  001204' 000000  000331
1392  001205' 000000  000000
1393                  000012          22180           ERRDD==.
1394                                  22200           DCF"/0"
1395  001204' 000000  000104          22220           DCL"DIVISION BY ZERO"
1396  001205' 000000  000111
1397  001206' 000000  000126
1398  001207' 000000  000111
1399  001210' 000000  000123
1400  001211' 000000  000111
1401  001212' 000000  000117
1402  001213' 000000  000116
1403  001214' 000000  000040
1404  001215' 000000  000102
1405  001216' 000000  000131
1406  001217' 000000  000040
1407  001220' 000000  000132
1408  001221' 000000  000105
1409  001222' 000000  000122
1410  001223' 000000  000117
1411  001223' 000000  000317
1412  001224' 000000  000000
1413                  000013          22240           ERRDV0==.
1414                                  22260           DCF"IL"
1415  001225' 000000  000111          22280           DCL"ILLEGAL DIRECT"
1416  001226' 000000  000114
1417  001227' 000000  000124
1418  001230' 000000  000105
1419  001231' 000000  000107
1420  001232' 000000  000101
1421  001233' 000000  000114
1422  001234' 000000  000040
1423  001235' 000000  000104
1424  001236' 000000  000111
```

```
1425  001237' 000000  000122
1426  001240' 000000  000105
1427  001241' 000000  000103
1428  001242' 000000  000124
1429  001243' 000000  000024
1430  001245' 000000  000000
1431                  000014          22300           ERRID==6
1432                                  22320   IFN     $TRING,<
1433                                  22340           DCE"TM"
1434  001244' 000000  000124         22360           DCL"TYPE MISMATCH"
1435  001245' 000000  000131
1436  001246' 000000  000120
1437  001247' 000000  000105
1438  001250' 000000  000040
1439  001251' 000000  000115
1440  001252' 000000  000111
1441  001253' 000000  000123
1442  001254' 000000  000115
1443  001255' 000000  000101
1444  001256' 000000  000124
1445  001257' 000000  000103
1446  001260' 000000  000110
1447  001260' 000000  000310
1448  001261' 000000  000000
1449                  000015          22380           ERRIM==5
1450                                  22400           DCE"OS"
1451  001262' 000000  000117         22420           DCL"OUT OF STRING SPACE"
1452  001263' 000000  000125
1453  001264' 000000  000124
1454  001265' 000000  000040
1455  001266' 000000  000117
1456  001267' 000000  000106
1457  001270' 000000  000040
1458  001271' 000000  000123
1459  001272' 000000  000124
1460  001273' 000000  000122
1461  001274' 000000  000111
1462  001275' 000000  000116
1463  001276' 000000  000107
1464  001277' 000000  000040
1465  001300' 000000  000123
1466  001301' 000000  000126
1467  001302' 000000  000101
1468  001303' 000000  000103
1469  001304' 000000  000105
1470  001304' 000000  000305
1471  001305' 000000  000000
1472                  000016          22440           ERRSO==5
1473                                  22460           DCE"LS"
1474  001306' 000000  000123         22480           DCL"STRING TOO LONG"
1475  001307' 000000  000124
1476  001310' 000000  000122
1477  001311' 000000  000111
```



```
1478  001312' 000000  000116
1479  001313' 000000  000107
1480  001314' 000000  000040
1481  001315' 000000  000124
1482  001316' 000000  000117
1483  001317' 000000  000117
1484  001320' 000000  000040
1485  001321' 000000  000114
1486  001322' 000000  000117
1487  001323' 000000  000116
1488  001324' 000000  000107
1489  001324' 000000  000307
1490  001325' 000000  000000
1491                  000017          22500           ERRLS==6
1492                                  22520           DCE"SI"
1493  001326' 000000  000123         22540           DCL"STRING FORMULA TOO COMPLEX"
1494  001327' 000000  000124
1495  001330' 000000  000122
1496  001331' 000000  000111
1497  001332' 000000  000116
1498  001333' 000000  000107
1499  001334' 000000  000040
1500  001335' 000000  000106
1501  001336' 000000  000117
1502  001337' 000000  000122
1503  001340' 000000  000115
1504  001341' 000000  000125
1505  001342' 000000  000114
1506  001343' 000000  000101
1507  001344' 000000  000040
1508  001345' 000000  000124
1509  001346' 000000  000117
1510  001347' 000000  000117
1511  001350' 000000  000040
1512  001351' 000000  000103
1513  001352' 000000  000117
1514  001353' 000000  000115
1515  001354' 000000  000120
1516  001355' 000000  000114
1517  001356' 000000  000105
1518  001357' 000000  000130
1519  001357' 000000  000330
1520  001360' 000000  000000
1521                  000020          22560           ERRST==0>
1522                                  22580   IFN     LENGTH,<
1523                                  22600           DCE"CN"
1524  001361' 000000  000103         22620           DCL"CAN'T CONTINUE"
1525  001362' 000000  000101
1526  001363' 000000  000116
1527  001364' 000000  000047
1528  001365' 000000  000124
1529  001366' 000000  000040
1530  001367' 000000  000103
```

```
 1531  001370' 000000  000117
 1532  001371' 000000  000116
 1533  001372' 000000  000124
 1534  001373' 000000  000111
 1535  001374' 000000  000116
 1536  001375' 000000  000125
 1537  001376' 000000  000105
 1538  001376' 000000  000305
 1539  001377' 000000  000000
 1540                  000021          22640           ERRCN==.-1
 1541                                  22660   IFN     FUNCTS,<
 1542                                  22680           DEFBUF"
 1543  001400' 000000  000125         22700           DCL"UNDEFINED USER FUNCTION"
 1544  001401' 000000  000116
 1545  001402' 000000  000104
 1546  001403' 000000  000105
 1547  001404' 000000  000106
 1548  001405' 000000  000111
 1549  001406' 000000  000116
 1550  001407' 000000  000105
 1551  001410' 000000  000104
 1552  001411' 000000  000040
 1553  001412' 000000  000125
 1554  001413' 000000  000123
 1555  001414' 000000  000105
 1556  001415' 000000  000122
 1557  001416' 000000  000040
 1558  001417' 000000  000106
 1559  001420' 000000  000125
 1560  001421' 000000  000116
 1561  001422' 000000  000103
 1562  001423' 000000  000124
 1563  001424' 000000  000111
 1564  001425' 000000  000117
 1565  001426' 000000  000116
 1566  001426' 000000  000316
 1567  001427' 000000  000000
 1568                  000022          22720           ERR.F==.-0>
 1569
 1570                                  22760   PAGE
```



```
 1571                                  22780   SUBTTL  LOW SEGMENT -- RAM -- IE THIS STUFF IS NOT CONSTANT
 1572                                  22800   ;
 1573                                  22820   ; THIS IS THE "VOLATILE" STORAGE AREA AND NONE OF IT
 1574                                  22840   ; CAN BE KEPT IN ROM, ANY CONSTANTS IN THIS AREA CANNOT
 1575                                  22860   ; BE KEPT IN A ROM, BUT MUST BE LOADED IN BY THE
 1576                                  22880   ; PROGRAM INSTRUCTIONS IN ROM.
 1577                                  22900   ;
 1578
 1579  001430' 000040  000054         22940   BUFMIN: 44                              ;A COMMA (PRELOAD OR ROM)
 1580                                  22960                                           ;USED BY INPUT STATEMENT SINCE THE
 1581                                  22980                                           ;DATA POINTER ALWAYS STARTS ON A
 1582                                  23000                                           ;COMMA OR TERMINATOR
 1583  001431'                         23020   BUF:    BLOCK   BUFLEN                  ;TYPE IN STORED HERE
 1584                                  23040                                           ;DIRECT STATEMENTS EXECUTE OUT OF
 1585                                  23060                                           ;HERE, REMEMBER INPUT SMASHES BUF,
 1586                                  23080                                           ;MUST BE AT A LOWER ADDRESS
 1587                                  23100                                           ;THAN DSCTMP OR ASSIGMENT OF STRING
 1588                                  23120                                           ;VALUES IN DIRECT STATEMENTS WON'T COPY
 1589                                  23140                                           ;INTL STRING SPACE -- WHICH IT MUST
 1590                                  23160   IFN     LPTSW,<
 1591                                  23180   LPTPOS: BLOCK   1                       ;POSITION OF LPT PRINT HEAD
 1592                                  23200   PRTFLG: BLOCK   1>                      ;WHETHER OUTPUT GOES TO LPT
 1593                                  23220                                           ;NON-ZERO MEANS SEND OUTPUT TO LPT
 1594                                  23240   IFN     CONTRW,<
 1595  001541'                         23260   CNTWFL: BLOCK   1>                      ;SUPRESS OUTPUT FLAG
 1596  001542'                         23280   DIMFLG: BLOCK   1                       ;IN GETTING A POINTER TO A VARIABLE
 1597                                  23300                                           ;IT IS IMPORTANT TO REMEMBER WHETHER IT
 1598                                  23320                                           ;IS BEING DONE FOR "DIM" OR NOT
 1599                                  23340                                           ;DIMFLG AND VALTYP MUST BE
 1600                                  23360                                           ;CONSECUTIVE LOCATIONS
 1601                                  23380   IFN     STRING,<
 1602  001543'                         23400   VALTYP: BLOCK   1                       ;THE TYPE INDICATOR
 1603                                  23420                                           ;IN THE 8K 0=NUMERIC 1=STRING
 1604  001544'                         23435   OPRTYP:                                 ;USED TO STORE OPERATOR NUMBER
 1605                                  23436                                           ;IN THE EXTENDED MOMENTARILY BEFORE
 1606                                  23437                                           ;OPERATOR APPLICATION
 1607  001544'                         23440   DONES:  BLOCK   1                       ;WHETHER CAN OR CAN'T CRUNCH RESRD WORDS
 1608                                  23460                                           ;TURNED ON IN THE 8K WHEN "DATA"
 1609                                  23480                                           ;BEING SCANNED BY CRUNCH SO UNQUOTED
 1610                                  23500                                           ;STRINGS WON'T BE CRUNCHED.
 1611  001545'                         23520   MEMSIZ: BLOCK   2                       ;HIGHEST LOCATION IN MEMORY
 1612  001547'                         23540   TEMPPT: BLOCK   2                       ;POINTER AT FIRST FREE TEMP DESCRIPTOR
 1613                                  23560                                           ;INITIALIZED TO POINT TO TEMPST
 1614  001551'                         23580   TEMPST: BLOCK   STRSIZ*NUMTMP           ;STORAGE FOR NUMTMP TEMP DESCRIPTORS
 1615  001570'                         23600   DSCTMP: BLOCK   STRSIZ                  ;STRING FUNCTIONS BUILD ANSWER DESCRIPTOR HERE
 1616  001575'                         23620   FRETOP: BLOCK   2>                      ;TOP OF STRING FREE SPACE
 1617                                  23640   IFN     LENGTH!STRING,<
 1618  001575'                         23660   TEMP3:  BLOCK   2>                      ;USED TO HOLD VARB # OF HIGH LOC FOUND
 1619                                  23680                                           ;IN GARBAGE COLLECTION
 1620                                  23700                                           ;AND USED MOMENTARILY BY FRNEVL
 1621                                  23720                                           ;USED IN EXTENDED BY FOUT
 1622                                  23740                                           ;ARRAY VARIABLE HANDLING TEMPORARY
 1623                                  23760   IFN     LENGTH,<
```

```
                                23780   DATLIN: BLOCK   2               ;DATA LINE # -- REMEMBER FOR ERRORS
                                23800   SUBFLG: BLOCK   1>              ;FLAG WHETHER SUBSCRIPTED VARIABLE ALLOWED
                                23820                                   ;"FOR" AND USER-DEFINED FUNCTION
                                23840                                   ;POINTER FETCHING TURN
                                23860                                   ;THIS ON BEFORE CALLING PTRGET
                                23880                                   ;SO ARRAYS WON'T BE DETECTED.
                                23900                                   ;STKINI AND PTRGET CLEAR IT.
                                23920   FLGINP: BLOCK   1               ;FLAGS WHETHER WE ARE DOING INPUT
                                23940                                   ;OR A READ
                                23960   TEMP1:  BLOCK   2               ;TEMPORARY FOR STATEMENT CODE
                                23980                                   ;NEWSTT SAVES [H,L] HERE FOR INPUT AND ^C
                                24000                                   ;"LET" SAVES NUMBERIC VARIABLE
                                24020                                   ;POINTERS HERE FOR "FOR"
                                24040                                   ;"NEXT" SAVES ITS TEXT POINTER HERE
                                24060                                   ;CLEARC SAVE [H,L] HERE
                                24080   TEMP2:  BLOCK   2               ;FORMULA EVALUATOR TEMP
                                24100                                   ;MUST BE PRESERVED BY OPERATORS
                                24120                                   ;USED IN EXTENDED BY FOUT
                                24140                                   ;ARRAY VARIABLE HANDLER TEMPORARY
                                24160   CURLIN: BLOCK   2               ;CURRENT LINE #
                                24180                                   ;SET TO 65535 WHEN DIRECT STATEMENTS EXECUTE
                                24200   IFN     LENGTH,<
                                24220   OLDLIN: BLOCK   2               ;OLD LINE NUMBER
                                24240   OLDTXT: BLOCK   2>              ;OLD TEXT POINTER
                                24260                                   ;POINTS AT STATEMENT TO BE EXECUTED NEXT
                                24280   STKTOP: BLOCK   2               ;TOP LOCATION TO USE FOR THE STACK
                                24300                                   ;INITIALLY SET UP BY INIT
                                24320                                   ;ACCORDING TO MEMORY SIZE
                                24340                                   ;TO ALLOW FOR 50 BYTES OF STRING SPACE.
                                24360                                   ;CHANGED BY A CLEAR COMMAND WITH
                                24380                                   ;AN ARGUMENT.
                                24400   TXTTAB: BLOCK   2               ;POINTER TO BEGINNING OF TEXT
                                24420                                   ;DOESN'T CHANGE AFTER BEING
                                24440                                   ;SETUP BY INIT.
                                24460   VARTAB: BLOCK   2               ;POINTER TO START OF SIMPLE
                                24480                                   ;VARIABLE SPACE
                                24500                                   ;UPDATED WHENEVER THE SIZE OF THE
                                24520                                   ;PROGRAM CHANGES, SET TO [TXTTAB]
                                24540                                   ;BY SCRATCH ("NEW").
                                24560   ARYTAB: BLOCK   2               ;POINTER TO BEGINNING OF ARRAY
                                24580                                   ;TABLE
                                24600                                   ;INCREMENTED BY 6 WHENEVER
                                24620                                   ;A NEW SIMPLE VARIABLE IS FOUND, AND
                                24640                                   ;SET TO [VARTAB] BY CLEARC.
                                24660   STREND: BLOCK   2               ;END OF STORAGE IN USE
                                24680                                   ;INCREASED WHENEVER A NEW ARRAY
                                24700                                   ;OR SIMPLE VARIABLE IS ENCOUNTERED
                                24720                                   ;SET TO [VARTAB] BY CLEARC.
                                24740   DATPTR: BLOCK   2               ;POINTER TO DATA, INITIALIZED TO POINT
                                24760                                   ;AT THE ZERO IN FRONT OF [TXTTAB]
                                24780   IFE     LENGTH=2,<
                                24800   TRCFLG: BLOCK   1>              ;0 MEANS NO TRACE IN PROGRESS
```



```
                                24840   ;THE FLOATING ACCUMULATOR
                                24860   IFE     LENGTH=2,<
                                24880           BLOCK   1               ;[TEMPORARY LEAST SIGNIFICANT BYTE]
                                24900   DFACLO: BLOCK   4>              ;[FOUR LOWEST ORDERS FOR DOUBLE PRECISION]
                                24920   FACLO:  BLOCK   3               ;[LOW ORDER OF MANTISSA]
                                24940                                   ;[MIDDLE ORDER OF MANTISSA]
                                24960                                   ;[HIGH ORDER OF MANTISSA]
                                24980   FAC:    BLOCK   2               ;[EXPONENT]
                                25000                                   ;[TEMPORARY COMPLEMENT OF SIGN IN MSB]
                                25020   IFE     LENGTH=2,<
                                25040           BLOCK   1               ;[TEMPORARY LEAST SIGNIFICANT BYTE]
                                25060   ARGLO:  BLOCK   7               ;[LOCATION OF SECOND ARGUMENT FOR DOUBLE
                                25080   ARG:    BLOCK   1>              ; PRECISION]
                                25100   FBUFFR: BLOCK   13              ;BUFFER FOR FOUT
                                25120   IFE     LENGTH=2,<BLOCK 35-13>
                                25140   PAGE
```

```
                                        25140   SUBTTL  TEXT CONSTANTS FOR PRINT OUT
                                        25160   ;
                                        25180   ; NEEDED FOR MESSAGES IN ALL VERSIONS
                                        25200   ; MUST BE STORED ABOVE OSCTMP OR ELSE STRLIT
                                        25220   ; WILL COPY THEM BEFORE START PRINTS THEM, THIS IS BAD, SINCE IF THE
                                        25240   ; USER IS OUT OF STRING SPACE BASIC WILL LOOP GETTING "OUT OF STRING SPACE"
                                        25260   ; ERRORS.
                                        25280   ;
                                        25300   IFN     LENGTH-2,<
                                        25320   ERR:    DC" ERROR"
                                        25340   0>
001720' 000040  000040          25360   INTXT:  DC" IN "
001721' 000040  000111
001722' 000000  000116
001723' 000000  000040
001724' 000000  000000          25380           0
001725' 000000  000015          25400   READY:  ACRLF
001726' 000000  000012
001727' 000000  000117          25420           DC"OK"
001730' 000000  000113
001731' 000000  000015          25440           ACRLF
001732' 000000  000012
001733' 000000  000000          25460           0
                                25480   IFN     LENGTH,<
001734' 000000  000015          25500   BRKTXT: ACRLF
001735' 000000  000012
001736' 000000  042102          25520           DC"BREAK"
001737' 000000  020122
001740' 000000  020105
001741' 000000  000101
001742' 000000  000113
001743' 000000  000313
001745' 000000  000000          25540           0>
                                25560   PAGE
```



```
                                        25600   SUBTTL  GENERAL STORAGE MANAGEMENT ROUTINES
                                        25620   ;
                                        25640   ; FIND A FOR ENTRY ON THE STACK WITH THE VARIABLE POINTER
                                        25660   ; PASSED IN [D,E].
                                        25680   ;
001744' 041000  000041          25700   FNDFOR: LXI     H,4+$CODE       ;IGNORING EVERYONES "NEWSTT"
001745' 000000  000004'
001746' 000000  000720'
                                        25720                           ;AND THE RETURN ADDRESS OF THIS
001747' 001000  000071          25740           DAD     SP              ;SUBROUTINE, SET [H,L]=SP
001750' 001000  000176          25760   LOOPER: MOV     A,M             ;SEE WHAT TYPE OF THING IS ON THE STACK
001751' 001000  000043          25780           INX     H
001752' 001000  000376          25800           CPI     FORTK           ;IS THIS STACK ENTRY A FOR?
001753' 000000  000201
001754' 001000  000300          25820           RNZ                     ;NO SO OK
                                25840   IF6     LENGTH,<
                                25860           PUSHM                   ;GET VARIABLE NAME
                                25880           XTML>
                                25900   IFN     LENGTH,<
001755' 001000  000116          25920           MOV     C,M
001756' 001000  000043          25940           INX     H               ;DO EQUIVALENT OF PUSHM / XTML
001757' 001000  000106          25960           MOV     B,M
001760' 001000  000043          25980           INX     H
001761' 001000  000345          26000           PUSH    H               ;PUT H  ON
001762' 001000  000151          26020           MOV     L,C             ;PUSH B / XTML IS SLOWER
001763' 001000  000140          26040           MOV     H,B
001764' 001000  000172          26060           MOV     A,D             ;FOR THE "NEXT" STATMENT WITHOUT AN ARGUMENT
001765' 001000  000263          26080           ORA     E               ;WE MATCH ON ANYTHING
001766' 001000  000353          26100           XCHG                    ;MAKE SURE WE RETURN [D,E]
001767' 001000  000312          26120           JZ      POP6OF          ;POINTING TO THE VARIABLE
001770' 000000  001774'
001771' 000000  001745'
001772' 001000  000353          26140           XCHG>
001773' 001000  000347          26160           COMPAM
001774' 001000  000001          26180   POP6OF: LXI     B,$CODE+13      ;TO WIPE OUT A FOR ENTRY
001775' 000000  000015'
001776' 000000  001770'
001777' 001000  000341          26200           POP     H
002000' 001000  000310          26220           RZ                      ;IF FOR MATCHES GOOD
002001' 001000  000011          26240           DAD     B
002002' 001000  000303          26260           JMP     LOOPER          ;TRY THE NEXT ONE
002003' 000000  001750'
002004' 000000  001775'
                                26280   ;
                                26300   ; THIS IS THE BLOCK TRANSFER ROUTINE
                                26320   ; IT MAKES SPACE BY SHOVING EVERYTHING FORWARD
                                26340   ;
                                26360   ; [H,L] = DESTINATION OF HIGH ADDRESS
                                26380   ; [D,E] = LOW ADDRESS TO BE TRANSFERRED
                                26400   ; [B,C] = HIGH ADDRESS TO BE TRANSFERRED
                                26420   ;
                                26440   ; A CHECK IS MADE TO MAKE SURE A REASONABLE AMOUNT
                                26460   ; OF SPACE REMAINS BETWEEN THE TOP OF THE STACK AND THE HIGHEST LOCATION
```

```
                                        26480   ; TRANSFERRED INTO
                                        26500   ;
                                        26520   ; ON EXIT [H,L]=[D,E]=LOW (B,C)=LOCATION LOW WAS MOVED INTO
                                        26540   ;
                                        26560
002005' 001000  000315          26580   BLTUI:  CALL    REASON          ;CHECK DESTINATION TO MAKE
002006' 002000  002045'
002007' 000000  002603'
                                        26600                           ;SURE THE STACK WON'T BE OVERRUN
002010' 001000  000305          26620   BLTUC:  PUSH    B               ;EXCHANGE [B,C] AND [H,L]
002011' 001000  000343          26640           XTHL
002012' 001000  000301          26660           POP     B
002013' 001000  000347          26680   BLTLOP: COMPAR                  ;SEE IF WE ARE DONE
002014' 001000  000176          26700           MOV     A,M             ;GET THE WORD TO TRANSFER
002015' 001000  000002          26720           STAX    B               ;TRANSFER IT
002016' 001000  000310          26740           RZ
002017' 001000  000013          26760           OCX     B
002020' 001000  000053          26780           OCX     H               ;BACKUP FOR NEXT GUY
002021' 001000  000303          26800           JMP     BLTLOP
002022' 000000  002613'
002023' 000000  002000'
                                        26820   ;
                                        26840   ; THIS ROUTINE IS USED TO MAKE SURE A CERTAIN NUMBER
                                        26860   ; OF LOCATIONS REMAIN AVAILABLE FOR THE
                                        26880   ; STACK.  THE CALL IS :
                                        26900   ;       CALL    GETSTK
                                        26920   ;       NUMBER OF 2 BYTE ENTRIES NECESSARY
                                        26940   ;
                                        26960   ; THIS ROUTINE MUST BE CALLED BY ANY ROUTINE WHICH PUTS
                                        26980   ; AN ARBITRARY AMOUNT OF STUFF ON THE STACK
                                        27000   ; (I.E. ANY RECURSIVE ROUTINE LIKE PARSE.)
                                        27020   ; IT IS ALSO CALLED BY ROUTINES SUCH AS GOSUB AND FOR
                                        27040   ; WHICH MAKE PERMANENT ENTRIES ON THE STACK
                                        27060   ;
                                        27080   ;
                                        27100   ; ROUTINES WHICH MERELY USE AND FREE UP THE GUARANTEED
                                        27120   ; NUMLEV STACK LOCATIONS NEED NOT CALL THIS
                                        27140   ;
002024' 001000  000343          27160   GETSTK: XTHL
002025' 001000  000116          27180           MOV     C,M             ;GET ARGUMENT INTO (C)
                                        27200
002026' 001000  000043          27220           INX     H
002027' 001000  000343          27240           XTHL                    ;PUT BACK RETURN ADDRESS
002030' 001000  000345          27260           PUSH    H               ;SAVE [H,L]
002031' 001000  000052          27280           LHLD    $TRENO
002032' 000000  001625'
002033' 000000  002422'
002034' 001000  000006          27300           MVI     B,0
002035' 000000  000000
002036' 001000  000011          27320           DAD     B
002037' 001000  000011          27340           DAD     B               ;SEE IF WE CAN HAVE THIS MANY
002040' 001000  000315          27360           CALL    REASON
002041' 000000  002605'
```



```
002042' 000000  002632'
002043' 001000  000341          27380           POP     H               ;RESTORE [H,L]
002044' 001000  000311          27400           RET
                                        27420   ;
                                        27440   ;
                                        27460   ; [H,L]= SOME ADDRESS
                                        27480   ; [H,L] IS EXAMINED TO MAKE SURE AT LEAST NUMLEV
                                        27500   ; LOCATIONS REMAIN BETWEEN IT AND THE TOP OF THE STACK
                                        27520   ;
002045' 001000  000325          27540   REASON: PUSH    D               ;SAVE [D,E]
002046' 001000  000353          27560           XCHG                    ;PUT [H,L] IN [D,E]
002047' 001000  000041          27600           LXI     H,$CODE+65536-2*NUMLEV  ;SETUP OFFSET OF GUARANTEED
002050' 000000  177726'
002051' 000000  002641'
                                        27620                           ;LOCATIONS
002052' 001000  000071          27640           DAD     SP              ;[H,L]=STACK POINTER + OFFSET
002053' 001000  000347          27660           COMPAR                  ;SEE IF THIS IS "GT" ENTERING [H,L]
002054' 001000  000353          27680           XCHG                    ;RESTORE [H,L] FROM [D,E]
002055' 001000  000321          27700           POP     D               ;GET [D,E] BACK
002056' 001000  000320          27720           RNC                     ;WAS OK?
002057' 001000  000036          27740   QMERR:  MVI     E,ENROM         ;"OUT OF MEMORY"
002060' 000000  000007
                                        27760   IFE     LENGTH,<
                                        27780           XWD     "D1000,1>       ;"LXI B," OVER THE NEXT 2
                                        27800   IFN     LENGTH,<
002061' 001000  000303          27820           JMP     ERROR>
002062' 000000  0x210c'
002063' 000000  002656'
                                        27840   PAGE
```

```
                                        27860  SUBTTL ERROR HANDLER, READY, COMPACTIFICATION, NEW, CLEAR, MAIN
                                        27880  IFN    LENGTH,<
002064' 001000  000652                  27900  DATSNE: LHLD   DATLIN          ;GET DATA LINE
002065' 000000  001557'
002066' 000000  002666'
002067' 001000  000642                  27920          SHLD   CURLIN>         ;MAKE IT CURRENT LINE
002070' 000000  001607'
002071' 000000  002665'
002072' 001000  000036                  27940  SNERR:  MVI    E,ERRSN         ;"SYNTAX ERROR"
002073' 000000  000002
002074' 001000  000001                  27960          XWD    ^O1000,1        ;"LXI B," OVER THE NEXT 2
002075' 001000  000036                  27980  OVERR:  MVI    E,ERROV#        ;DIVISION BY ZERO
002076' 000000  000015
                                        28000  IFN    LENGTH,<
002077' 001000  000001                  28020          XWD    ^O1000,1        ;SKIP NEXT TWO
002100' 001000  000036                  28040  NFERR:  MVI    E,ERWNF>        ;"NEXT WITHOUT FOR" ERROR
002101' 000000  000001
002102' 001000  000315                  28060  ERROR:  CALL   STKINI          ;RESET THE STACK AND FLAGS
002103' 000000  002470'
002104' 000000  002670'
                                        28080  IFN    CONTRW,<
002105' 001000  000257                  28100          XRA    A
002106' 001000  000062                  28120          STA    CNTWFL>         ;FORCE OUTPUT
002107' 000000  001541'
002110' 000000  002103'
002111' 001000  000315                  28140          CALL   CRDO            ;CRLF
002112' 000000  004523'
002113' 000000  002107'
002114' 001000  000041                  28220          LXI    H,ERRTAB        ;GET START OF ERROR TABLE
002115' 000000  000730'
002116' 000000  002116'
                                        28240  IFE    LENGTH-2,<
002117' 001000  000315                  28260  LEPSKP: CALL   REM             ;SKIP AN ERROR MESSAGE
002120' 000000  004674'
002121' 000000  002115'
002122' 001000  000035                  28280          DCR    E               ;DECREMENT ERROR COUNT
002123' 001000  000043                  28282          INX    H               ;SKIP OVER THIS ERROR MESSAGE
002124' 001000  000302                  28302          JNZ    LEPSKP>         ;SKIP SOME MORE
002125' 000000  002117'
002126' 000000  002122'
                                        28320  IFN    LENGTH-2,<
                                        28322          MOV    D,A             ;GET ZERO INTO D
                                        28324          MVI    A,"?"           ;START OF ERROR MESSAGE
                                        28326          OUTCHR                 ;TYPE IT
                                        28340          DAD    D               ;ADD IN ERROR CODE
                                        28360          MOV    A,M             ;GET FIRST ERROR CHARACTER
                                        28380          OUTCHR                 ;TYPE IT
                                        28400          CHRGET                 ;GET 2ND CHARACTER OF ERROR CODE
                                        28420          OUTCHR                 ;TYPE IT
                                        28440          LXI    H,ERR>          ;GET POINTER TO " ERROR"
002127' 001000  000315                  28460  ERRFIN: CALL   STROUT          ;TYPE IT
002130' 000000  007743'
002131' 000000  002125'
```



```
002132' 001000  000652                  28480          LHLD   CURLIN          ;CURRENT LINE #
002133' 000000  001607'
002134' 000000  002130'
002135' 001000  000174                  28500          MOV    A,H             ;SEE IF IN DIRECT MODE
002136' 001000  000245                  28520          ANA    L
002137' 001000  000074                  28540          INR    A               ;ZERO SAYS DIRECT MODE
002140' 001000  000304                  28560          CNZ    INPRT           ;PRINT LINE NUMBER IN (H,L)
002141' 000000  000000*
002142' 000000  002133'
                                        28580  IFE    LENGTH,<
                                        28600          XWD    ^O1000,1        ;"LXI B," OVER THE NEXT 2
                                        28620  ENDI:
                                        28640  STOP:   RNZ                    ;MAKE SURE WE TERMINATED IT
                                        28660  STPEND: POP    B               ;GET RID OF "NEWSTT" RETURN ADDRESS
                                        28680  ENDCON:>
                                        28682  IFE    LENGTH-2,<
                                        28684  ;
                                        28686  ; FOR "LIST" COMMAND STOPPING
                                        28688  ;
002143' 001000  000076                  28690          XWD    ^O1000,^O76             ;SKIP THE NEXT BYTE
002144' 001000  000301                  28692  STPRDY: POP    B>
                                        28700  READY:  IFN    LPTSW,<
                                        28720          CALL   FINLPT>         ;PRINT ANY LEFT OVERS
                                        28740  IFN    CONTRW,<
002145' 001000  000257                  28760          XRA    A
002146' 001000  000062                  28780          STA    CNTWFL>         ;FORCE OUTPUT
002147' 000000  001541'
002150' 000000  002141'
002151' 001000  000041                  28800          LXI    H,$CODE+65535
002152' 000000  177777'
002153' 000000  002147'
002154' 001000  000042                  28820          SHLD   CURLIN          ;SETUP CURLIN FOR DIRECT MODE
002155' 000000  001607'
002156' 000000  002152'
002157' 001000  000041                  28840          LXI    H,READY         ;"READY" CRLF CR,F
002160' 000000  001725'
002161' 000000  002155'
002162' 001000  000315                  28860  REPINI: CALL   INIT            ;PRINT IT, REPLACED BY CALL STROUT
002163' 000000  000000*
002164' 000000  002160'
                                        28880                                 ;BY THE INIT CODE, THIS IS HERE SO AFTER
                                        28900                                 ;ERRORS DURING INIT, INIT IS RESTARTED
002165' 001000  000315                  28920  MAIN:   CALL   INLIN           ;GET A LINE FROM TTY
002166' 000000  003776'
002167' 000000  002163'
002170' 001000  000327                  28940          CHRGET                 ;GET THE FIRST
002171' 001000  000074                  28960          INR    A               ;SEE IF 0 SAYING THE CARRY FLAG
002172' 001000  000075                  28980          DCR    A
002173' 001000  000312                  29000          JZ     MAIN            ;IF SO, A BLANK LINE WAS INPUT
002174' 000000  002165'
002175' 000000  002166'
002176' 001000  000365                  29020          PUSH   PSW             ;SAVE STATUS INDICATOR FOR 1ST CHARACTER
002177' 001000  000315                  29040          CALL   LINGET          ;READ IN A LINE #
```

```
BASIC MBS 8080  =ATBS/ALLEN/DAVIDOFF     MACRO 47(113) 03:12 10-SEP-75 PAGE 8-2
F3      MAC     6-SEP-64 03:11          ERROR HANDLER, READY, COMPACTIFICATION, NEW, CLEAR, MAIN

 1972  002200' 000000  003642'
 1973  002201' 000000  002174'
 1974  002202' 001000  000325          29060           PUSH    D               ;SAVE LINE #
 1975  002203' 001000  000315          29080           CALL    CRUNCH          ;CRUNCH THE LINE DOWN
 1976  002204' 000000  002533'
 1977  002205' 000000  002200'
 1978  002206' 001000  000107          29100           MOV     B,A             ;A=# AFTER CRUNCH, (B,C)=CHAR COUNT FOR NODEL
 1979  002207' 001000  000321          29120           POP     D               ;RESTORE LINE #
 1980  002210' 001000  000361          29140           POP     PSW             ;WAS THERE A LINE #?
 1981  002211' 001000  000322          29160           JNC     GONE            ;IF NOT IT'S A DIRECT STATEMENT
 1982  002212' 000000  003370'
 1983  002213' 000000  002204'
 1984  002214' 001000  000325          29180           PUSH    D
 1985  002215' 001000  000305          29200           PUSH    B               ;SAVE LINE # AND CHARACTER COUNT
 1986  002216' 001000  000327          29220           CHNGET                  ;REMEMBER IF THIS LINE IS
 1987  002217' 001000  000365          29240           PUSH    PSW             ;BLANK SO WE DON'T INSERT IT
 1988  002220' 001000  000315          29260           CALL    FNDLIN          ;GET A POINTER TO THE LINE
 1989  002221' 000000  002371'
 1990  002222' 000000  002212'
 1991  002223' 001000  000305          29280           PUSH    B               ;SAVE THE POINTER
 1992                                  29300   IFE     LENGTH=2,<
 1993  002224' 001000  000334          29320           CC      DEL>            ;DELETE THE LINE
 1994  002225' 000000  011271'
 1995  002226' 000000  002221'
 1996                                  29340   IFN     LENGTH=2,<
 1997                                  29360           JNC     NODEL           ;NO MATCH SO DON'T DELETE
 1998                                  29380           XCHG                    ;(D,E) NOW HAS THE POINTER TO THE LINE
 1999                                  29400           ;BEYOND THIS ONE
 2000                                  29420           LHLD    VARTAB          ;COMPACTIFYING TO VARTAB
 2001                                  29440   MLOOP:  LDAX    D
 2002                                  29460           STAX    B               ;SHOVING DOWN TO ELIMINATE A LINE
 2003                                  29480           INX     B
 2004                                  29500           INX     D
 2005                                  29520           COMPAR
 2006                                  29540           JNC     MLOOP           ;DONE COMPACTIFYING?
 2007                                  29560           MOV     H,B
 2008                                  29580           MOV     L,C
 2009                                  29600           INX     H               ;NEW VARTAB
 2010                                  29620           SHLD    VARTAB>
 2011  002227' 001000  000321          29640   NODEL:  POP     D               ;POP POINTER AT PLACE TO INSERT
 2012  002230' 001000  000361          29660           POP     PSW             ;SEE IF THIS LINE HAD
 2013                                  29680           ;ANYTHING ON IT
 2014  002231' 001000  000314          29700           JZ      FINI            ;IF NOT DON'T INSERT
 2015  002232' 000000  002276'
 2016  002233' 000000  002265'
 2017  002234' 001000  000052          29720           LHLD    VARTAB          ;CURRENT END
 2018  002235' 000000  001621'
 2019  002236' 000000  002232'
 2020  002237' 001000  000343          29740           XTHL                    ;(H,L)=CHARACTER COUNT, VARTAB
 2021                                  29760           ;ONTO THE STACK
 2022  002240' 001000  000301          29780           POP     B               ;(B,C)=OLD VARTAB
 2023  002241' 001000  000011          29800           DAD     B
 2024  002242' 001000  000345          29820           PUSH    H               ;SAVE NEW VARTAB
```

```
BASIC MBS 8080  =ATBS/ALLEN/DAVIDOFF     MACRO 47(113) 03:12 10-SEP-75 PAGE 8-3
F3      MAC     6-SEP-64 03:11          ERROR HANDLER, READY, COMPACTIFICATION, NEW, CLEAR, MAIN

 2025  002243' 001000  000315          29840           CALL    BLTU
 2026  002244' 000000  002605'
 2027  002245' 000000  002235'
 2028  002246' 001000  000341          29860           POP     H               ;POP OFF VARTAB
 2029  002247' 001000  000044          29880           SHLD    VARTAB          ;UPDATE VARTAB
 2030  002250' 000000  001621'
 2031  002251' 000000  002244'
 2032  002252' 001000  000353          29900           XCHG
 2033  002253' 001000  000164          29920           MOV     M,H             ;FOOL CHEAD WITH NON-ZERO LINK
 2034  002254' 001000  000043          29940           INX     H               ;SO IT DOESN'T THINK
 2035                                  29960           ;THIS LINK IS THE
 2036                                  29980           ;END OF THE PROGRAM
 2037  002255' 001000  000043          30000           INX     H
 2038  002256' 001000  000321          30020           POP     D               ;GET LINE # OFF STACK
 2039  002257' 001000  000163          30040           MOV     M,E
 2040  002260' 001000  000043          30060           INX     H               ;PUT DOWN LINE #
 2041  002261' 001000  000162          30080           MOV     M,D
 2042  002262' 001000  000043          30100           INX     H
 2043
 2044  002263' 001000  000021          30140           LXI     D,BUF           ;MOVE LINE FROM BUF TO PROGRAM AREA
 2045  002264' 000000  001431'
 2046  002265' 000000  002256'
 2047  002266' 001000  000032          30160   MLOOPN: LDAX    D               ;NOW TRANSFERING LINE
 2048                                  30180           ;IN FROM BUF
 2049  002267' 001000  000167          30200           MOV     M,A
 2050  002270' 001000  000043          30220           INX     H
 2051  002271' 001000  000023          30240           INX     D
 2052  002272' 001000  000267          30260           ORA     A               ;ZERO MARKS THE END
 2053  002273' 001000  000362          30280           JNZ     MLOOPN
 2054  002274' 000000  002266'
 2055  002275' 000000  002264'
 2056  002276' 001000  000315          30300   FINI:   CALL    RUNC            ;DO CLEAR & SET UP STACK
 2057  002277' 000000  002437'
 2058  002300' 000000  002274'
 2059                                  30320           ;ALSO SETS (H,L) TO (TXTTAB)-1
 2060  002301' 001000  000043          30340           INX     H
 2061                                  30360   ;
 2062                                  30380   ; CHEAD GOES THROUGH PROGRAM STORAGE AND FIXES
 2063                                  30400   ; UP ALL THE LINKS, THE END OF EACH
 2064                                  30420   ; LINE IS FOUND BY SEARCHING FOR THE ZERO AT THE END,
 2065                                  30440   ; THE DOUBLE ZERO LINK IS USED TO DETECT THE END OF THE PROGRAM
 2066                                  30460   ;
 2067  002302' 001000  000124          30480   CHEAD:  MOV     D,H             ;(D,E)=(H,L)
 2068  002303' 001000  000135          30500           MOV     E,L
 2069  002304' 001000  000176          30520           MOV     A,M             ;SEE IF END OF CHAIN
 2070  002305' 001000  000043          30540           INX     H               ;BUMP POINTER
 2071  002306' 001000  000266          30560           ORA     M               ;2ND BYTE
 2072  002307' 001000  000312          30580           JZ      MAIN            ;DONE
 2073  002310' 000000  002165'
 2074  002311' 000000  002277'
 2075  002312' 001000  000043          30600           INX     H               ;FIX H TO START OF TEXT
 2076  002313' 001000  000043          30620           INX     H
 2077  002314' 001000  000043          30640           INX     H
```

```
2078  002315' 001000  000257          30660          XRA     A               ;SEARCHING FOR A ZERO IN MEMORY
2079  002316' 001000  000176          30680  CZLOOP: CMP     M               ;TO MARK THE END OF THIS LINE
2080  002317' 001000  000043          30700          INX     H               ;BUMP POINTER
2081  002320' 001000  000302          30720          JNZ     CZLOOP          ;END OF LINE
2082  002321' 000000  002316'
2083  002324' 000000  002310'
2084  002323' 001000  000353          30740          XCHG                    ;SWITCH TEMP
2085  002324' 001000  000163          30760          MOV     M,E             ;DO FIRST BYTE OF FIXUP
2086  002325' 001000  000043          30780          INX     H               ;ADVANCE POINTER
2087  002326' 001000  000162          30800          MOV     M,D             ;2ND BYTE OF FIXUP
2088  002327' 001000  000353          30820          XCHG                    ;AND BACK AGAIN
2089  002330' 001000  000303          30840          JMP     CMEAD           ;KEEP CHAINING TIL DONE
2090  002331' 000000  002310'
2091  002332' 000000  002321'
2092                                  30860  IFE     LENGTH-2,<
2093                                  30880  ;
2094                                  30900  ; SCNLIN SCANS A LINE RANGE OF
2095                                  30920  ; THE FORM  #-# OR # OR #- OR -# OR BLANK
2096                                  30940  ; AND THEN FINDS THE FIRST LINE IN THE RANGE
2097                                  30960  ;
2098  002333' 001000  000021          30980  SCNLIN: LXI     D,$CODE         ;ASSUME START LIST AT ZERO
2099  002334' 000000  000000'
2100  002335' 000000  002331'
2101  002336' 001000  000325          31000          PUSH    D               ;SAVE INITIAL ASSUMPTION
2102  002337' 001000  000312          31020          JZ      ALLLST          ;IF FINISHED, LIST IT ALL
2103  002340' 000000  002354'
2104  002341' 000000  002334'
2105  002342' 001000  000321          31040          POP     D               ;WE ARE GOING TO GRAB A #
2106  002343' 001000  000315          31060          CALL    LINGET          ;GET A LINE #, IF NONE, RETURNS ZERO
2107  002344' 000000  003640'
2108  002345' 002000  002340'
2109  002346' 001000  000425          31080          PUSH    D               ;SAVE FIRST
2110  002347' 001000  000312          31100          JZ      ONELIN          ;IF ONLY # THEN DONE.
2111  002350' 000000  002365'
2112  002351' 000000  002344'
2113  002352' 001000  000317          31120          SYNCHK  MINUTK          ;MUST BE A DASH,
2114  002353' 000000  000051
2115  002354' 001000  000021          31140  ALLLST: LXI     D,$CODE+^O65529 ;ASSUME MAX END OF RANGE
2116  002355' 000000  177771'
2117  002356' 000000  002354'
2118  002357' 001000  000304          31160          CNZ     LINGET          ;GET THE END OF RANGE
2119  002360' 000000  003640'
2120  002361' 000000  002353'
2121  002362' 001000  000302          31180          JNZ     SNERR           ;MUST BE TERMINATOR
2122  002363' 000000  002267'
2123  002364' 000000  002360'
2124  002365' 001000  000353          31200  ONELIN: XCHG                    ;[H,L] = FINAL
2125  002366' 001000  000321          31220          POP     D               ;GET INITIAL IN [D,E]
2126  002367' 001000  000343          31240          XTHL                    ;PUT MAX ON STACK, RETURN ADDR TO [H,L]
2127  002370' 001000  000345          31260          PUSH    H               ;SAVE RETURN ADDRESS BACK
2128                                  31280  ;
2129                                  31300  ; FNDLIN SEARCHES THE PROGRAM TEXT FOR THE LINE
2130                                  31320  ; WHOSE LINE # IS PASSED IN [D,E]. [D,E] IS PRESERVED.
```



```
2131                                  31340  ; THERE ARE THREE POSSIBLE RETURNS:
2132                                  31360  ;
2133                                  31380  ;       1) ZERO FLAG SET, CARRY NOT SET.  LINE NOT FOUND.
2134                                  31400  ;          NO LINE IN PROGRAM GREATER THAN ONE SOUGHT,
2135                                  31420  ;          [B,C] POINTS TO TWO ZERO BYTES AT END OF PROGRAM,
2136                                  31440  ;          [H,L]=[B,C]
2137                                  31460  ;
2138                                  31480  ;       2) ZERO, CARRY SET.
2139                                  31500  ;          [B,C] POINTS TO THE LINK FIELD IN THE LINE
2140                                  31520  ;          WHICH IS THE LINE SEARCHED FOR,
2141                                  31540  ;          [H,L] POINTS TO THE LINK FIELD IN THE NEXT LINE.
2142                                  31560  ;
2143                                  31580  ;       3) NON-ZERO, CARRY NOT SET,
2144                                  31600  ;          LINE NOT FOUND, [B,C]  POINTS TO LINE IN PROGRAM
2145                                  31620  ;          GREATER THAN ONE SEARCHED FOR,
2146                                  31640  ;          [H,L] POINTS TO THE LINK FIELD IN THE NEXT LINE.
2147                                  31660  ;
2148  002371' 001000  000052          31680  FNDLIN: LHLD    TXTTAB          ;GET POINTER TO START OF TEXT
2149  002372' 000000  001617'
2150  002373' 000000  002363'
2151  002374' 001000  000104          31700  LOOP:   MOV     B,H             ;IF EXITING BECAUSE OF END OF PROGRAM,
2152                                  31720                                  ;SET [B,C] TO POINT TO DOUBLE ZEROES.
2153  002375' 001000  000115          31740          MOV     C,L
2154  002376' 001000  000176          31760          MOV     A,M             ;GET WORD POINTER TO
2155  002377' 001000  000043          31780          INX     H               ;BUMP POINTER
2156  002400' 001000  000266          31800          ORA     M               ;GET 2ND BYTE
2157  002401' 001000  000053          31820          DCX     H               ;GO BACK
2158  002402' 001000  000310          31840          RZ                      ;IF ZERO THEN DONE
2159  002403' 001000  000505          31860          PUSH    B
2160  002404' 001000  000367          31880          PUSHM                   ;PUSH LINK
2161  002405' 001000  000367          31900          PUSHM                   ;PUSH BINARY LINE #
2162  002406' 001000  000341          31920          POP     H               ;POP HERE
2163  002407' 001000  000347          31940          COMPAR                  ;COMPARE [D,E] TO [H,L]
2164  002410' 001000  000341          31960          POP     H               ;GET LINK
2165  002411' 001000  000301          31980          POP     B               ;GET POINTER TO THIS LINE IN [B,C]
2166  002412' 001000  000277          32000          CMC                     ;TURN CARRY ON
2167  002413' 001000  000310          32020          RZ                      ;EQUAL RETURN
2168  002414' 001000  000277          32040          CMC                     ;MAKE CARRY ZERO
2169  002415' 001000  000320          32060          RNC                     ;NO MATCH RETURN (GREATER)
2170  002416' 001000  000303          32080          JMP     LOOP            ;KEEP LOOPING
2171  002417' 000000  002374'
2172  002420' 000000  002372'
2173                                  32100  ;
2174                                  32120  ; THE "NEW" COMMAND CLEARS THE PROGRAM TEXT AS WELL
2175                                  32140  ; AS VARIABLE SPACE
2176                                  32160  ;
2177  002421' 001000  000300          32180  SCRATH: RNZ                     ;MAKE SURE THERE IS A TERMINATOR
2178  002422' 001000  000052          32200  SCRTCH: LHLD    TXTTAB
2179  002423' 000000  001617'
2180  002424' 000000  002417'
2181                                  32220  IFN     LENGTH-2,<
2182                                  32240          XRA     A>
2183                                  32260  IFE     LENGTH-2,<
```

```
2184  002425' 001000  000315          32280            CALL    TUFF>           ;TURN OFF TRACE, SET (A)=0,
2185  002426' 000000  003605'
2186  002427' 000000  002623'
2187  002430' 001000  000167          32300            MOV     M,A             ;SAVE AT END OFF TEXT
2188  002431' 001000  000043          32320            INX     H               ;BUMP POINTER
2189  002432' 001000  000167          32340            MOV     M,A             ;SAVE ZERO
2190  002433' 001000  000043          32360            INX     H               ;BUMP POINTER
2191  002434' 001000  000042          32380            SHLD    VARTAB          ;NEW START OF VARIABLE
2192  002435' 000000  001621'
2193  002436' 000000  002426'
2194                                   32400    IFE     LENGTH,<
2195                                   32420    RUN1:   NAZ>
2196  002437' 001000  000052          32440    RUNC:   LHLD    TXTTAB          ;POINT AT THE START OF TEXT
2197  002440' 000000  001617'
2198  002441' 000000  002435'
2199  002442' 001000  000053          32460            DCX     H
2200                                   32480    ;
2201                                   32500    ; CLEARC IS A SUBROUTINE WHICH INITIALIZES THE VARIABLE AND
2202                                   32520    ; ARRAY SPACE BY RESETING ARYTAB (THE END OF SIMPLE VARIABLE SPACE)
2203                                   32540    ; AND STREND (THE END OF ARRAY STORAGE), IT FALLS INTO STKINI
2204                                   32560    ; WHICH RESETS THE STACK, (H,L) IS PRESERVED.
2205                                   32580    ;
2206                                   32600            IFE     STRING,<CLEAR1>
2207  002443' 001000  000042          32620    CLEARC: SHLD    TEMP            ;SAVE (H,L) IN TEMP
2208  002444' 000000  001603'
2209  002445' 000000  002446'
2210                                   32640            IFN     STRING,<
2211  002440' 001000  000052          32660            LHLD    MEMSIZ
2212  002447' 000000  001545'
2213  002450' 000000  002444'
2214  002451' 001000  000042          32680            SHLD    FRETOP>         ;FREE UP STRING SPACE
2215  002452' 000000  001573'
2216  002453' 000000  002447'
2217  002454' 001000  000315          32700            CALL    RESTORE         ;RESTORE DATA
2218  002455' 000000  003446'
2219  002456' 000000  00245E'
2220  002457' 001000  000052          32720            LHLD    VARTAB          ;GET START OF VARIABLE SPACE
2221  002460' 000000  001621'
2222  002461' 000000  002455'
2223  002462' 001000  000042          32740            SHLD    ARYTAB          ;SAVE IN START OF ARRAY SPACE
2224  002463' 000000  001623'
2225  002464' 000000  00246D'
2226  002465' 001000  000042          32760            SHLD    STREND          ;AND END OF VARIABLE STORAGE
2227  002466' 000000  001625'
2228  002467' 000000  00246B'
2229                                   32780    ;
2230                                   32800    ; STKINI RESETS THE STACK POINTER ELIMINATING
2231                                   32820    ; GOSUB & FOR CONTEXT.  STRING TEMPORARIES ARE FREED
2232                                   32840    ; UP, SUBFLG IS RESET, CONTINUING IS DISALLOWED,
2233                                   32860    ; AND A DUMMY ENTRY IS PUT ON THE STACK, THIS IS SO
2234                                   32880    ; FNDFOR WILL ALWAYS FIND A NON-"FOR" ENTRY AT THE BOTTOM
2235                                   32900    ; OF THE STACK, (A)=0 AND (D,E) IS PRESERVED.
2236                                   32920    ;
```



```
2237  002470' 001000  000301          32940    STKINI: POP     B               ;GET RETURN ADDRESS HERE
2238  002471' 001000  000052          32960            LHLD    STKTOP          ;(H,L) POINTER TO END OF MEMORY
2239  002472' 000000  001615'
2240  002473' 000000  002466'
2241  002474' 001000  000371          32980            SPHL                    ;INITIALIZE STACK
2242                                   33000    IFN     STRING,<
2243  002475' 001000  000041          33020            LXI     H,TEMPST
2244  002476' 000000  001551'
2245  002477' 000000  002476'
2246  002500' 001000  000042          33040            SHLD    TEMPPT>         ;INITIALIZE STRING TEMPORARIES
2247  002501' 000000  001547'
2248  002502' 000000  002476'
2249  002503' 001000  000041          33060            LXI     H,SCOVE         ;PUT ZERO (NON NEXT, FOR, GOSUB TOKEN)
2250  002504' 000000  000000'
2251  002505' 000000  002501'
2252  002506' 001000  000345          33080            PUSH    H               ;ON STACK
2253  002507' 001000  000042          33100    IFN     LENGTH,<SHLD    OLDTXT> ;MAKE CONTINUING ILLEGAL
2254  002510' 000000  001613'
2255  002511' 000000  002504'
2256                                   33120    IFN     LPTSW,<
2257                                   33140            CALL    FINLPT>
2258  002512' 001000  000052          33160            LHLD    TEMP            ;GET SAVED (H,L)
2259  002513' 000000  001603'
2260  002514' 000000  002510'
2261                                   33180    IFN     LENGTH,<
2262                                   33200    IFE     CONTRW,<XRA     A>
2263  002515' 001000  000062          33220            STA     SUBFLG>         ;ALLOW SUBSCRIPTS
2264  002516' 000000  001601'
2265  002517' 000000  002513'
2266  002520' 001000  000305          33240            PUSH    B               ;PUT RETURN ADDRESS BACK ON
2267  002521' 001000  000311          33260            RET                     ;GO BACK
2268
2269  002522' 001000  000076          33300    QINLIN: MVI     A,"?"           ;GET A QMARK
2270  002523' 000000  000077
2271  002524' 001000  000337          33320            OUTCHR                  ;TYPE IT
2272  002525' 001000  000076          33340            MVI     A," "           ;SPACE
2273  002526' 000000  000040
2274  002527' 001000  000337          33360            OUTCHR                  ;TYPE IT TOO
2275                                   33380    IFE     STRING,<CALL    INLIN   ;IN THE NON-STRING VERSIONS ALL
2276                                   33400                                    ;INPUT IS CRUNCHED
2277                                   33420            INX     H>              ;SET A LINE OF INPUT FROM TTY
2278  002530' 001000  000303          33440    IFN     STRING,<JMP     INLIN>  ;NO CRUNCHING IN THIS CASE
2279  002531' 000000  002276'
2280  002532' 000000  00251B'
2281                                   33460    ;
2282                                   33520    ; ALL "RESERVED" WORDS ARE TRANSLATED INTO SINGLE
2283                                   33540    ; BYTES WITH THE MSB ON. THIS SAVES SPACE AND TIME
2284                                   33560    ; BY ALLOWING FOR TABLE DISPATCH DURING EXECUTION.
2285                                   33580    ; THEREFORE ALL STATEMENTS APPEAR TOGETHER IN THE
2286                                   33600    ; RESERVED WORD LIST IN THE SAME
2287                                   33620    ; ORDER THEY APPEAR IN IN STMDSP.
2288                                   33640    ;
2289  002535'                          33700    CRUNCH: IFN     STRING,<
```

```
2290  002533' 001000  000E57          33720           XRA     A
2291  002534' 001000  000862          33740           STA     DORES>          ;ALLOW CRUNCHING
2292  002537' 000000  0A1544'
2293  002538' 000000  002531'
2294  002537' 001000  000015          33780           MVI     C,5             ;COUNT OF CHARS AT LEAST 5
2295  002540' 000000  000005
2296  002541' 001000  000021          33760           LXI     D,BUF           ;SETUP DESTINATION POINTER
2297  002542' 000000  0A1431'
2298  002543' 000000  002535'
2299  002544' 001000  000176          33800   KLOOP:  MOV     A,M             ;GET CHARACTER FROM BUF
2300  002545' 001000  000376          33820           CPI     " "             ;IS IT A SPACE WE WANT TO SAVE
2301  002546' 000000  000040
2302  002547' 001000  000312          33840           JZ      STUFFM          ;YES, STUFF IN DESTINATION LINE,
2303  002550' 000000  002607'
2304  002551' 000000  002546'
2305  002552' 001000  000107          33860           MOV     B,A             ;GET A CHARACTER FROM THE LINE
2306                                   33880                                   ;SETUP B WITH A QUOTE IF IT IS A STRING
2307  002553' 001000  000376          33900           CPI     34              ;QUOTE SIGN?
2308  002554' 000000  000042
2309  002555' 001000  000312          33920           JZ      STRNG           ;YES, GO TO SPECIAL STRING HANDLING
2310  002556' 000000  002721'
2311  002557' 000000  002550'
2312  002560' 001000  000267          33940           ORA     A               ;END OF LINE?
2313  002561' 001000  000312          33960           JZ      CRDONE          ;YES, DONE CRUNCHING
2314  002562' 000000  0A2753'
2315  002563' 000000  002550'
2316                                   33980   IFN     STRING,<
2317  002564' 001000  000267          34000           ORA     DORES           ;IN DATA STATEMENT AND NO CRUNCH?
2318  002565' 000000  021544'
2319  002566' 000000  002562'
2320  002567' 001000  000267          34020           ORA     A
2321  002570' 001000  000107          34040           MOV     B,A             ;INITIALIZE RESERVED WORD COUNT
2322  002571' 001000  000176          34060           MOV     A,M             ;GET THE CHARACTER AGAIN
2323  002572' 001000  000302          34080           JNZ     STUFFM>         ;IF NO CRUNCHING JUST STORE
2324  002573' 000000  002667'
2325  002574' 000000  002565'
2326                                   34100                                   ;THE CHARACTER
2327                                   34120   IFN     LENGTH,<                ;NOT FOR 4K VERSION
2328  002575' 001000  000376          34140           CPI     "Z"             ;A MARK?
2329  002576' 000000  0A0E77
2330  002577' 001000  000076          34160           MVI     A,PRINTK
2331  002600' 000000  000031
2332  002601' 001000  000312          34180           JZ      STUFFM          ;THEN USE A "PRINT" TOKEN
2333  002602' 000000  002667'
2334  002603' 000000  002573'
2335  002604' 001000  000176          34200           MOV     A,M
2336  002605' 001000  000376          34220           CPI     "0"             ;SKIP NUMERICS
2337  002606' 000000  000060
2338  002607' 001000  000332          34240           JC      MUSTCR          ;SINCE CRUNCHING IS SLOW
2339  002610' 000000  002611'
2340  002611' 000000  002602'
2341  002612' 001000  000376          34260           CPI     60              ;":" ALSO PUT IN QUICKLY
2342  002613' 000000  000074
```



```
2343  002614' 001000  000332          34280           JC      STUFFM
2344  002615' 000000  002667'
2345  002616' 000000  002614'
2346  002617'                          34300   MUSTCR: B
2347  002617' 001000  000325          34320           PUSH    D               ;SAVE STORE POINTER
2348                                   34340   IFE     STRING,<
2349                                   34360           MVI     B,0>            ;INIT RESERVED WORD COUNT
2350  002620' 001000  000021          34380           LXI     D,RES_ST-1      ;INIT TO RESERVED WORD LIST
2351  002621' 000000  000171'
2352  002622' 000000  002613'
2353  002623' 001000  000345          34400           PUSH    H               ;SAVE IN BUF POINTER
2354  002624' 001000  021000          34420           XWD     "C1000,"076     ;"MVI" AROUND CHRGET
2355  002625' 001000  000327          34440   NXTRES: CHRGET                  ;GET CHAR FROM LINE
2356  002626' 001000  000023          34460           INX     D               ;BUMP DEPOSIT POINTER
2357  002627' 001000  000032          34480   RESER:  LDAX    D               ;GET A BYTE FROM RESERVED WORD LIST
2358  002630' 001000  000346          34500           ANI     127             ;GET RID OF SIGN BIT
2359  002631' 000000  000177
2360  002632' 001000  000312          34520           JZ      TABEND          ;END OF RESERVED WORD TABLE
2361  002633' 000000  002653'
2362  002634' 000000  002621'
2363  002635' 001000  000276          34540           CMP     M               ;TWO CHARS THE SAME?
2364  002636' 001000  000302          34560           JNZ     NTHIS           ;NO, DIFFERENT
2365  002637' 000000  002730'
2366  002640' 000000  002633'
2367  002641' 001000  000032          34580           LDAX    D               ;GET RESERVED WORD BYTE
2368  002642' 001000  027267          34600           ORA     A               ;SET CONDITION CODES
2369  002643' 001000  000362          34620           JP      NXTRES          ;IF SIGN SET, RESERVED WORD FOUND
2370  002644' 000000  002626'
2371  002645' 000000  002637'
2372  002646' 001000  000361          34640   FOUND:  POP     PSW             ;TAKE OFF GARBAGE ORIG POINTER
2373  002647' 001000  000170          34660           MOV     A,B             ;GET RESERVED WORD #
2374  002650' 001000  000366          34680           ORI     128             ;SET MSB TO FLAG AS RESERVED WORD
2375  002651' 000000  000200
2376  002652' 001000  000362          34700           XWD     "C1000,"C362    ;"JP" AROUND THE POP H AND MOV A,M
2377  002653' 001000  000341          34720   TABEND: POP     H               ;GET BACK ORIG POINTER
2378  002654' 001000  000176          34740           MOV     A,M             ;GET BACK ORIG CHAR
2379  002655' 001000  000321          34760           POP     D               ;GET STUFF POINTER BACK
2380                                   34780   IFE     LENGTH-2,<
2381  002656' 001000  000353          34800           XCHG                    ;(H,L)=STUFF POINTER
2382  002657' 001000  000376          34820           CPI     ELSETK          ;HAVE TO PUT A HIDDEN
2383  002660' 000000  000020
2384                                   34840                                   ;COLON IN FRONT OF "ELSE"S
2385  002661' 001000  000066          34860           MVI     M,":"           ;STORE IT
2386  002662' 000000  000072
2387  002663' 001000  000314          34880           CZ      INXHRT##        ;ADVANCE POINTER ON "ELSE"
2388  002664' 000000  000000*
2389  002665' 000000  002644'
2390                                   34900                                   ;SO ONLY ON "ELSE" THE COLON IS NOT OVERWRITTEN
2391
2392  002666' 001000  000353          34920           XCHG>                   ;(D,E)=STUFF POINTER
2393  002667' 001000  000043          34940   STUFFM: INX     H               ;ENTRY TO BUMP (H,L)
2394  002670' 001000  000022          34960           STAX    D               ;SAVE CHARACTER IN CRUNCHED LINE
2395  002671' 001000  000023          34980           INX     D               ;BUMP SAVE POINTER
```

```
 2396  002670' 001000  000019          35000           INX     B               ;BUMP CRUNCHED CHARACTER COUNT
 2397                                  59020   IFN     STRING,<
 2398  002673' 001000  000326          35040           SUI     ":"             ;SEE IF IT IS A COLON
 2399  002674' 000000  000072
 2400  002675' 001000  000312          35060           JZ      COLIS           ;IF SO ALLOW CRUNCHING AGAIN
 2401  002676' 000000  002765'
 2402  002677' 000000  002664'
 2403  002700' 001000  000376          35080           CPI     DATATK-"I"
 2404  002701' 000000  000111
 2405  002702' 001000  000302          35100           JNZ     NODATT
 2406  002703' 000000  002710'
 2407  002704' 000000  002676'
 2408  002705' 001000  000066          35120   COLIS:  STA     COLRES          ;SETUP FLAG
 2409  002706' 000000  001344'
 2410  002707' 000000  002703'
 2411  002710' 001000  000326          35140   NODATT: SUI     REMTK-"I"&"
 2412  002711' 000000  000124
 2413                                  35160   IFE     STRING,<SUI     REMTK>  ;WAS IT A REM STATEMENT
 2414  002712' 001000  000302          35180           JNZ     KLOOP           ;KEEP LOOPING
 2415  002713' 000000  002544'
 2416  002714' 000000  002706'
 2417  002715' 001000  000307          35200           MOV     B,A             ;REM DOESN'T STOP ON "I", ONLY ON A ZERO
 2418  002716' 001000  000176          35220   STR1:   MOV     A,M             ;GET A CHAR
 2419  002717' 001000  000267          35240           ORA     A               ;SET CONDITION CODES
 2420  002720' 001000  000312          35260           JZ      CRDONE          ;IF END OF LINE THEN DONE
 2421  002721' 000000  002753'
 2422  002722' 000000  002713'
 2423  002723' 001000  000270          35280           CMP     B               ;END OF SUBSTR
 2424  002724' 001000  000312          35300           JZ      STUFFM          ;IF YES, DONE WITH STRING
 2425  002725' 000000  002661'
 2426  002726' 000000  002721'
 2427  002727' 001000  000043          35320   STRNG1  INX     H               ;INCREMENT TEXT POINTER
 2428  002730' 001000  000022          35340           STAX    D               ;STORE CHAR
 2429  002731' 001000  000014          35360           INR     C               ;BUMP COUNT
 2430  002732' 001000  000025          35380           INX     D               ;AND POINTER
 2431  002733' 001000  000303          35400           JMP     STR1            ;KEEP LOOPING
 2432  002734' 000000  002716'
 2433  002735' 000000  002725'
 2434
 2435  002736' 001000  000041          35440   NTHIS1  POP     H               ;RESTORE TEXT STRING
 2436  002737' 001000  000345          35460           PUSH    H               ;AND SAVE IT BACK
 2437  002740' 001000  000004          35480           INR     B               ;INCREMENT RESERVED WORD #
 2438  002741' 001000  000353          35500           XCHG                    ;RESLST POINTER INTO (H,L)
 2439  002742' 001000  000166          35520   NTHIS1: ORA     M               ;TEST BITS IN RESERVED WORD LIST
 2440  002743' 001000  000043          35540           INX     H
 2441  002744' 001000  000366          35560           JP      NTHIS1          ;SKIP MORE
 2442  002745' 000000  002740'
 2443  002746' 000000  002734'
 2444  002747' 001000  000353          35580           XCHG                    ;RESLST POINTER INTO (D,E)
 2445                                  35600                   ;TEXT POINTER INTO (H,L)
 2446  002750' 001000  000303          35620           JMP     RESER           ;DONE, MOVE TO NEXT RESERVED WORD
 2447  002751' 000000  002621'
 2448  002752' 000000  002745'
```



```
 2449  002753' 001000  000041          35640   CRDONE: LXI     H,BUFMIN        ;LEAVE WITH (H,L) POINTER TO START OF LINE
 2450  002754' 000000  001430'
 2451  002755' 000000  002751'
 2452  002756' 001000  000042          35660           STAX    D               ;NEED THREE 0'S ON THE END
 2453  002757' 001000  000025          35680           INX     D               ;ONE FOR END-OF-LINE
 2454  002760' 001000  000022          35700           STAX    D               ;AND 2 FOR A ZERO LINK
 2455  002761' 001000  000023          35720           INX     D               ;SINCE IF THIS IS A DIRECT STATEMENT
 2456  002762' 001000  000066          35740           STAX    D               ;ITS END MUST LOOK LIKE THE END OF A PROGRAM
 2457  002763' 001000  000311          35760           RET                     ;END OF CRUNCHING
 2458                                  35780   ;
 2459                                  35800   ; THIS IS THE LINE INPUT ROUTINE
 2460                                  35820   ; IT READS CHARACTERS INTO BUF USING _ AS THE
 2461                                  35840   ; CHARACTER DELETE CHARACTER AND @ AS THE LINE DELETE CHARACTER
 2462                                  35860   ; IF MORE THAN BUFLEN CHARACTER ARE TYPED, NO ECHOING
 2463                                  35880   ; IS DONE UNTIL A _ OR @ OR CARRIAGE-RETURN IS TYPED.
 2464                                  35900   ; CONTROL-G WILL BE TYPED FOR EACH EXTRA CHARACTER.
 2465                                  35920   ; THE ROUTINE IS ENTERED AT INLIN
 2466                                  35940   ;
 2467  002764' 001000  000005          35960   LINLIN: DCR     B               ;BACK ARROW SO DECREMENT COUNT
 2468  002765' 001000  000053          35980           DCX     H               ;BALK UP POINTER
 2469                                  36000   IFN     REALIO,<
 2470  002766' 001000  000337          36020           OUTCHR>
 2471  002767' 001000  000302          36040           JNZ     INLINC          ;NOT TOO MANY SO CONTINUE
 2472  002770' 000000  003103'
 2473  002771' 000000  002754'
 2474  002772' 000000  002772'        36060   INLINN: IFN     REALIO,<
 2475  002772' 001000  000137          36080           OUTCHR>                 ;PRINT THE @, OR A SECOND _ IF THERE
 2476                                  36100                   ;WERE TOO MANY
 2477  002773' 001000  000315          36120           CALL    CRDO            ;TYPE A CRLF
 2478  002774' 000000  002523'
 2479  002775' 000000  002770'
 2480  002776' 001000  000041          36140   INLINI: LXI     H,BUF
 2481  002777' 000000  001431'
 2482  003000' 000000  002774'
 2483  003001' 001000  000006          36160           MVI     B,1             ;CHARACTER COUNT
 2484  003002' 000000  000001
 2485  003003' 001000  000315          36180   INLINC: CALL    INCHR           ;GET A CHARACTER
 2486  003004' 000000  003126'
 2487  003005' 000000  002777'
 2488                                  36182   IFN     LENGTH,<
 2489  003006' 001000  000376          36184           CPI     7               ;IS IT BOB ALBRECHT RINGING THE BELL
 2490  003007' 000000  000007
 2491  003010' 001000  000312          36186           JZ      GOODCH>         ;FOR SCHOOL KIDS?
 2492  003011' 000000  003044'
 2493  003012' 000000  003004'
 2494  003013' 001000  000376          36200           CPI     15              ;IS IT A CARRIAGE RETURN?
 2495  003014' 000000  000015
 2496  003015' 001000  003312          36220           JZ      FINLNL          ;IF SO FINISH UP
 2497  003016' 000000  002451'
 2498  003017' 000000  003011'
 2499  003020' 001000  000376          36240           CPI     32              ;CHECK FOR FUNNY CHARACTERS
 2500  003021' 000000  000040
 2501  003022' 001000  003032          36260           JC      INLINE
```

```
 2502  003025' 000000  003003'
 2503  003026' 000000  003016'
 2504  003027' 001000  000376          36280           CPI     125
 2505  003028' 000000  000175
 2506  003029' 001000  000322          36300           JNC     INLINC          ;BIG ONES BAD TOO
 2507  003030' 000000  003003'
 2508  003031' 000000  003023'
 2509  003032' 001000  000376          36320           CPI     "@"             ;LINE DELETE?
 2510  003033' 000000  000100
 2511  003034' 001000  000312          36340           JZ      INLINN
 2512  003035' 000000  00277E'
 2513  003036' 000000  003036'
 2514  003037' 001000  000376          36360           CPI     "_"             ;CHARACTER DELETE?
 2515  003040' 000000  000137
 2516  003041' 001000  000312          36380           JZ      LINLIN
 2517  003042' 000000  012,64'
 2518  003043' 000000  003035'
 2519  003044' 001000  000117          36400   GOJDCH: MOV     C,A
 2520  003045' 001000  000170          36420           MOV     A,B
 2521  003046' 001000  000376          36440           CPI     BUFLEN
 2522  003047' 000000  000110
 2523  003050' 001000  000076          36460           MVI     A,7             ;GET A BELL IN CASE LINE TOO LONG.
 2524  003051' 000000  000007
 2525  003052' 001000  000322          36480           JNC     OUTBEL          ;LINE TOO LONG, RING BELL.
 2526  003053' 000000  003061'
 2527  003054' 000000  003042'
 2528  003055' 001000  000171          36500           MOV     A,C
 2529  003056' 001000  000161          36520           MOV     M,C             ;STORE THIS CHARACTER
 2530  003057' 001000  000043          36540           INX     H
 2531  003060' 001000  000004          36560           INR     B
 2532  003061'                          36580   OUTBEL:
 2533                                   36600           IFN     REALIO,<
 2534  003061' 001000  000337          36620           OUTCHR>
 2535  003062' 001000  000303          36640           JMP     INLIN.
 2536  003063' 000000  003003'
 2537  003064' 000000  003053'
 2538  003065'                          36660   OUTLOW:
 2539                                   36680   IFN     CONTRW,<
 2540  003065' 001000  000302          36700           JNZ     PPSWRI>         ;NO, DO OUTPUT
 2541  003066' 000000  010024'
 2542  003067' 000000  003663'
 2543                                   36720   IFN     REALIO,<                ;WITH I/O
 2544                                   36740   IFN     LPTSW,<
 2545                                   36760           LDA     PRTFLG          ;SEE IF WE WANT TO TALK TO LPT
 2546                                   36780           ORA     A               ;TEST BITS
 2547                                   36800           JZ      TTYCHR          ;IF ZERO THEN NOT
 2548                                   36820           POP     PSW             ;GET CHARACTER WE WANT TO PRINT
 2549                                   36840           PUSH    PSW
 2550                                   36860           CPI     13              ;IS IT CARRIAGE RETURN?
 2551                                   36880           CZ      PRINTW          ;FORCE OUT A LINE
 2552                                   36900           JC      PPSWRI          ;IF FUNNY CONTROL CHARACTER (LF) DO NOTHING
 2553                                   36920           LDA     LPTPOS          ;WHERE ARE WE?
 2554                                   36940           CPI     LPTLEN          ;AT THE END?
```



```
 2555                                   36960           CNC     PRINTW          ;YES, START OVER
 2556                                   36980           INR     A
 2557                                   37000           STA     LPTPOS
 2558                                   37020   LPTWAT: IN      2
 2559                                   37040           ANI     2
 2560                                   37060           JZ      LPTWAT
 2561                                   37080           POP     PSW
 2562                                   37100           OUT     3               ;SEND OUT CHAR
 2563                                   37120           RET                     ;RETURN
 2564                                   37140   PATLPT: BLOCK   20
 2565                                   37160   FINLPT: XRA     A               ;RESET PRINT FLAG SO OUTPUT
 2566                                   37180           STA     PRTFLG          ;GOES TO THE TERMINAL
 2567                                   37200           LDA     LPTPOS          ;SEE IF ANY LEFTOVERS MUST BE
 2568                                   37220           ORA     A               ;FORCED OUT
 2569                                   37240           RZ                      ;BY LOOKING AT LPTPOS
 2570                                   37260   PRINTW: IN      2               ;MAKE SURE LAST PRINT
 2571                                   37280           ANI     2               ;FINISHED BY TESTING DONE
 2572                                   37300           JZ      PRINTW          ;BIT
 2573                                   37320   ; SEE IF BUFFER MUST BE EMPTIED
 2574                                   37340           LDA     LPTPOS
 2575                                   37360           ORA     A               ;CHARACTERS IN THE BUFFER?
 2576                                   37380           JNZ     PRINTR          ;IF SO DON'T CLEAR THE BUFFER
 2577                                   37400           MVI     A,4             ;OTHERWISE BUFFER MUST BE EMPTIED
 2578                                   37420           OUT     2               ;CLEAR THE BUFFER
 2579                                   37440                                   ;TO PRINT A BLANK LINE
 2580                                   37460   PRINTR: MVI     A,1             ;TELL LPT TO PRINT
 2581                                   37480           OUT     2               ;STATUS REG
 2582                                   37500           DCR     A               ;(A)=0
 2583                                   37520           STA     LPTPOS          ;RESET LINE PRINT POSITION
 2584                                   37540           RET>
 2585  003070'                          37560   TTYCHR:>
 2586                                   37580   IFN     STRING,<
 2587  003070' 001000  000561          37600           POP     PSW             ;GET THE CHARACTER
 2588  003071' 001000  000365          37620           PUSH    PSW             ;AND SAVE IT AGAIN
 2589  003072' 001000  000376          37640           CPI     32              ;IS THIS A MEANINGFUL CHARACTER?
 2590  003073' 000000  000040
 2591  003074' 001000  000332          37660           JC      TRYOUT>         ;IF IT'S A NON-PRINTING CHARACTER
 2592  003075' 000000  003115'
 2593  003076' 000000  003066'
 2594                                   37680                                   ;DON'T INCLUDE IT IN TTYPOS
 2595                                   37700   IFN     LENGTH!CONTRW,LPTSW,<
 2596  003077' 001000  000072          37720           LDA     TTYPOS>         ;SEE IF PRINT HEAD IS AT THE END OF THE LINE
 2597  003100' 000000  000047'
 2598  003101' 000000  003075'
 2599                                   37740           CPI     LINLEN          ;MODIFIED BY "TERMINAL WIDTH?" QUESTION IN INIT
 2600  003102' 001000  000376
 2601  003103' 000000  000110
 2602                                   003103'
 2603  003104' 001000  000314          37760   LINPT1==1.-1
 2604  003105' 000000  003213'                  CZ      CRDO            ;TYPE CRLF AND SET TTYPOS AND (A)=0 IF SO
 2605  003106' 000000  003106'
 2606  003107' 001000  000074          37800           INR     A               ;INCREMENT TTYPOS SINCE WE'RE
 2607                                   37820                                   ;GOING TO PRINT A CHARACTER.
```

```
2608  003110' 001000  000662          37840           STA     TTYPOS          ;STORE NEW PRINT HEAD POSITION
2609  003111' 000000  000047'
2610  003112' 000000  003105'
2611  003115'                         37860   TRYOUT:
2612                                  37880   IFN     REALIO,<
2613  003115' 001000  000333          37900   NOPRIN: IN      0               ;GET STATUS
2614  003116' 000000  000000
2615                                  003116' 37920   CNLLA1==.+1             ;CONSOLE COMMAND CHANGE LOC
2616  003115' 001000  000346          37940           ANI     IDONE           ;OK TO SEND CHAR
2617  003116' 000000  000000
2618  003117' 001000  000302          37960           JNZ     NOPRIN>         ;KEEP LOOPING
2619  003120' 000000  003113'
2620  003121' 000000  003111'
2621  003126' 001000  000361          37980           POP     PSW             ;GET CHARACTER BACK
2622  003125' 001000  000323          38000           OUT     TTOCHN          ;SEND OUT THE CHAR
2623  003124' 000000  000001
2624                                  003124' 38020   CNLLB1==.+1             ;CONSOLE COMMAND CHANGE LOC
2625  003125' 001000  000311          38040           RET                     ;RETURN FROM OUTCHR
2626
2627
2628  003126'                         38100   INCHR:  IFN     REALIO,<
2629  003126' 001000  000333          38120   TRYIN:  IN      0               ;GET STATUS
2630  003127' 000000  000000
2631                                  003127' 38140   CNLLA2==.+1             ;CONSOLE COMMAND CHANGE LOC
2632  003130' 001000  000346          38160           ANI     IDONE           ;TEST BIT
2633  003131' 000000  000001
2634  003132' 001000  000302          38180           JNZ     TRYIN>          ;GO BACK & DO IT AGAIN
2635  003133' 000000  003126'
2636  003134' 000000  003120'
2637  003135' 001000  000333          38200           IN      TTICHN          ;GET A CHAR
2638  003136' 000000  000001
2639                                  003136' 38220   CNLLB2==.+1             ;CONSOLE COMMAND CHANGE LOC
2640  003137' 001000  000346          38240           ANI     127             ;GET RID OF PARITY BIT
2641  003140' 000000  000177
2642                                  38260   IFN     CONTRL,<
2643  003141' 001000  000376          38280           CPI     CONTRL          ;IS IT SUPRESS OUTPUT?
2644  003142' 000000  000017
2645  003143' 001000  000300          38300           RNZ
2646  003144' 001000  000072          38320           LDA     CNTRFL
2647  003145' 000000  001341'
2648  003146' 000000  003135'
2649  003147' 001000  000057          38340           CMA                     ;COMPLEMENT ITS STATE
2650  003150' 001000  000062          38360           STA     CNTRFL>         ;SAVE BACK
2651  003151' 000000  001341'
2652  003152' 000000  003145'
2653  003155' 001000  000311          38380           RET
2654                                  38400   PAGE
```



```
2655                                  38420   SUBTTL  THE "LIST" COMMAND
2656
2657                                  38460   IFN     LENGTH-2,<
2658                                  38480   IFN     LPTSW,<
2659                                  38500   LLIST:  MVI     A,1             ;GET NON ZERO VALUE
2660                                  38520           STA     PRTFLG>         ;SAVE IN I/O FLAG
2661                                  38540   LIST:   CALL    LINGET          ;GET LINE NUMBER INTO (D,E)
2662                                  38560           RNZ                     ;MUST BE A TERMINATOR OR ERROR
2663                                  38580           POP     B               ;GET RID OF NEWSTT RETURN ADDR
2664                                  38600           CALL    FNDLIN          ;FIND LINE GREATER THAN OR EQUAL TO (D,E)
2665                                  38620           PUSH    B               ;SAVE START POINTER
2666                                  38640   LIST4:  POP     H               ;GET POINTER TO LINE
2667                                  38660           PUSHH                   ;PUSH LINK
2668                                  38680           POP     B               ;TAKE OFF FOR A SECOND
2669                                  38700           MOV     A,B             ;SEE IF END OF CHAIN
2670                                  38720           ORA     C
2671                                  38740           JZ      READY
2672                                  38760   IFN     LISTEN,<
2673                                  38780           CALL    ISCNTC>         ;CHECK FOR CONTROL-C
2674                                  38800           PUSH    B               ;PUT BACK ON
2675                                  38820           CALL    CRDO            ;DO CRLF TO START OUT
2676                                  38840           PUSHH                   ;PUSH LINE #
2677                                  38860           XTHL                    ;GET LINE # INTO (H,L)
2678                                  38880                                   ;AND WE WANT (H,L) ON THE STACK
2679                                  38900           CALL    LINPRT          ;PRINT AS INT WITOUT LEADING SPACE
2680                                  38920           MVI     A," "
2681                                  38940   PRIT4:  POP     H               ;RESTORE POINTER TO START OF TEXT
2682                                  38960           PLOOP:  OUTCHR          ;ALWAYS A SPACE AFTER THE LINE #
2683                                  38980           MOV     A,M             ;GET A CHARACTER FROM LINE.
2684                                  39000           ORA     A               ;IS IT A RESERVED WORD
2685                                  39020           INX     H               ;INCREMENT POINTER INTO TEXT
2686                                  39040           JZ      LIST4           ;ZERO, END OF LINE, GET NEXT LINE
2687                                  39060           JP      PLOOP           ;REGULAR CHAR, JUST PRINT IT
2688                                  39080           SUI     127             ;GET RID OF SIGN BIT AND ADD ONE
2689                                  39100           MOV     C,A             ;GET RESERVED WORD # IN C
2690                                  39120           PUSH    H               ;SAVE CURRENT POSIT
2691                                  39140           LXI     D,RESLST        ;GET RESLST POINTER.
2692                                  39160   RESNCM: PUSH    D               ;SAVE
2693                                  39180
2694                                  39200   RESCR1: LDAX    D               ;GET CHARACTER FROM RESLST
2695                                  39220           INX     D               ;BUMP RESLST POINTER
2696                                  39240           ORA     A               ;TEST BITS
2697                                  39260           JP      RESCR1          ;NOT AT END OF RESERVED WORD YET
2698                                  39280           DCR     C               ;DECREMENT CNAR
2699                                  39300           POP     H               ;POP START POINTER HERE
2700                                  39320           JNZ     RESNCM          ;NOT AT END OF RESLST YET.
2701                                  39340   ;HERE WHEN FOUND RIGHT RESERVED WORD
2702                                  39360   PRIT3:  MOV     A,M             ;GET A CHARACTER FROM RESERVED WORD
2703                                  39380           ORA     A               ;SET CONDITION CODES
2704                                  39400           JM      PRIT4
2705                                  39420           OUTCHR
2706                                  39440           INX     H               ;BUMP RESLST POINTER
2707                                  39460           JMP     PRIT3>          ;PRINT THE REST
```

```
                                        34520   PAGE
```



```
                                        39540   SUBTTL "FOR" STATEMENT
                                        39560   ;
                                        39580   ; NOTE:
                                        39600   ;
                                        39620   ; A FOR ENTRY ON THE STACK HAS THE FOLLOWING FORMAT:
                                        39640   ;
                                        39660   ;
                                        39680   ; LOW ADDRESS
                                        39700   ;       TOKEN (FORTK IN HIGH BYTE)  1 BYTE
                                        39720   ;       A POINTER TO THE LOOP VARIABLE  2 BYTES
                                        39740   ;       A BYTE REFLECTING THE SIGN OF THE INCREMENT 1 BYTE
                                        39760   ;       THE STEP 4 BYTES
                                        39800   ;       THE UPPER VALUE 4 BYTES
                                        39820   ;       THE LINE # OF THE "FOR" STATEMENT 2 BYTES
                                        39840   ;       A TEXT POINTER INTO THE "FOR" STATEMENT 2 BYTES
                                        39860   ; HIGH ADDRESS
                                        39880   ;
                                        39900   ; TOTAL 16 BYTES
                                        39920   ;
                                        39940
0031547'                                39960   FOR:    IFN     LENGTH,<
0031547' 001000  000076                 39980           MVI     A,100
0031552' 000000  000144
0031556' 001000  000006                 40000           STA     SUBFLG>         ;DONT RECOGNIZE SUBSCRIPTED VARIABLES
0031557' 000000  001001'
0031561' 000000  003151'
0031563' 001000  000315                 40020           CALL    LET             ;READ THE VARIABLE AND ASSIGN IT
0031566' 000000  000131'
0031570' 000000  003157'
                                        40040                                   ;THE CORRECT INTIAL VALUE
                                        40060                                   ;AND STORE A POINTER
                                        40080                                   ;TO THE VARIABLE IN (TEMP)
0031573' 001000  000343                 40100           XTHL                    ;SAVE TEXT PTR ON THE STACK
0031576' 001000  000315                 40120           CALL    FNDFOR          ;MUST HAVE VARIABLE POINTER IN (D,E)
0031601' 000000  001744'
0031605' 000000  003162'
0031610' 001000  000321                 40140           POP     D               ;(D,E)=TEXT POINTER
0031611' 001000  000302                 40160           JNZ     NOTOL           ;IF NO MATCHING ENTRY, DON'T
0031614' 0 00000 003176'
0031617' 000000  003166'
                                        40180                                   ;ELIMINATE ANYTHING
0031620' 001000  000011                 40200           DAD     B               ;IN THE CASE OF "FOR"
                                        40220                                   ;WE ELIMINATE THE MATCHING ENTRY
                                        40240                                   ;AS WELL AS EVERYTHING AFTER IT
0031623' 000000  000371                 40260           SPHL                    ;DO THE ELIMINATION
                                        40280                                   ;SINCE A MATCHING ENTRY WAS FOUND
0031626' 001000  000353                 40300   NOTOL:  XCHG                    ;(H,L)=TEXT POINTER
0031627' 001000  000315                 40320           CALL    GETSTK
0032030' 000000  002024'
0032033' 000000  003172'
0032036' 001000  000010                 40340           B                       ;MAKE SURE 16 BYTES ARE AVAILABLE
                                        40360                                   ;OFF OF THE STACK
0032039' 001000  000345                 40380           PUSH    H               ;REALLY SAVE THE TEXT POINTER
```

```
2764  003204' 001000  000315          40400           CALL    DATA            ;GET AN (M,L) THAT POINTS
2765  003205' 000000  000477'
2766  003206' 000000  0b5c00"
2767                                  40420                                   ;JUST BEYOND THE TERMINATOR
2768  003207' 001000  000343          40440           XTHL                    ;PUT (M,L) POINTER TO TERMINATOR ON THE STACK
2769                                  40460                                   ;AND RESTORE (M,L) AS TEXT POINTER AT
2770                                  40480                                   ;VARIABLE NAME
2771  003210' 001000  000345          40500           PUSH    H               ;PUSH THE TEXT POINTER ONTO THE STACK
2772  003211' 001000  000052'         40520           LHLD    CURLIN          ;(M,L) GET THE CURRENT LINE #
2773  003212' 000000  001607'
2774  003213' 000000  0b3c05'
2775  003214' 001000  000343          40540           XTHL                    ;NOW THE CURRENT LINE # IS ON THE STACK AND
2776                                  40560                                   ;(M,L) IS THE TEXT POINTER
2777                                  40580   IFN     LENGTH-2,<
2778                                  40600   IFN     STRING,<CALL    CHKNUM>>
2779  003215' 001000  000317          40620           SYNCHK  TOTK            ;"TO" IS NECESSARY
2780  003216' 000000  000c41
2781                                  40640   IFN     LENGTH-2,<
2782                                  40660           CALL    FRMNUM>         ;READ FINAL VALUE
2783                                  40680   IFE     LENGTH-2,<
2784  003217' 001000  000315          40700           CALL    FRMEVL>
2785  003220' 000000  0b533b'
2786  003221' 000000  003e1e'
2787  003222' 001000  000345          40720           PUSH    H               ;SAVE THE TEXT POINTER
2788                                  40740   IFE     LENGTH-2,<
2789  003223' 001000  000315          40760           CALL    FRCSNG>
2790  003224' 000000  00d670*
2791  003225' 000000  003e20'
2792  003226' 001000  000315          40780           CALL    MOVRF           ;GET THE STUFF
2793  003227' 000000  000000*
2794  003230' 000000  003e24'
2795  003231' 001000  000341          40800           POP     H               ;REGAIN TEXT POINTER
2796  003232' 001000  000305          40820           PUSH    B               ;OPPOSITE OF PUSH
2797  003233' 001000  000325          40840           PUSH    D               ;SAVE THE SIGN OF THE INCREMENT
2798  003234' 001000  000001          40860           LXI     B,$CODE+*0201+256
2799  003235' 000000  1c0400'
2800  003236' 000000  003e2f'
2801  003237' 001000  000121          40880           MOV     D,C
2802  003240' 001000  000132          40900           MOV     E,D             ;GET 1,0 IN THE REGISTERS
2803  003241' 001000  000176          40920           MOV     A,M             ;GET TERMINATING CHARACTER
2804  003242' 001000  000376          40940           CPI     STEPTK          ;DO WE HAVE "STEP" ?
2805  003243' 000000  000247
2806  003244' 001000  000076          40960           MVI     A,1             ;SETUP DEFAULT SIGN
2807  003245' 000000  000001
2808  003246' 001000  0b030e          40980           JNZ     ONEON           ;PUSH SOME CONSTANTS ON IF NOT
2809  003247' 000000  003c65'
2810  003250' 000000  003235'
2811                                  41000   IFN     LENGTH-2,<
2812                                  41020   IFN     STRING,<
2813                                  41040           CHRGET
2814                                  41060           CALL    FRMNUM>>        ;READ THE STEP
2815                                  41080   IFE     <LENGTH-2>&<STRING,<
2816  003251' 001000  000315          41100           CALL    FRMCHK>         ;DON'T NEED TO CHECK THE TYPE
```



```
2817  003252' 000000  0b533f'
2818  003253' 000000  003e47'
2819  003254' 001000  000345          41120           PUSH    H
2820                                  41140   IFE     LENGTH-2,<
2821  003255' 001000  000315          41160           CALL    FRCSNG>
2822  003256' 000000  003e24*
2823  003257' 000000  003e52'
2824  003260' 001000  000315          41180           CALL    MOVRF           ;SET UP THE REGISTERS
2825  003261' 000000  003e27*
2826  003262' 000000  003e56'
2827  003263' 001000  000341          41200           POP     H
2828  003264' 001000  000357          41220           FSIGN                   ;GET THE SIGN OF THE INCREMENT
2829  003265' 001000  000305          41240   ONEON:  PUSH    B               ;PUT VALUE ON BACKWARDS
2830  003266' 001000  000325          41260           PUSH    D               ;OPPOSITE OF PUSH
2831  003267' 001000  000365          41280   IFORON: PUSH    PSW             ;SAVE THE SIGN OF THE INCREMENT
2832  003270' 001000  000043          41300           INX     SP              ;A ONE BYTE ENTRY ONLY
2833  003271' 001000  000345          41320           PUSH    H
2834  003272' 001000  000052          41340           LHLD    TEMP            ;GET THE POINTER TO THE VARIABLE BACK
2835  003273' 000000  001603'
2836  003274' 000000  0b5c61'
2837  003275' 001000  000343          41360           XTHL                    ;PUT THE POINTER TO THE VARIABLE
2838                                  41380                                   ;ONTO THE STACK AND RESTORE THE TEXT POINTER
2839  003276' 001000  000006          41400   NXTCON: MVI     B,FORTK         ;PUT A "FOR" TOKEN ONTO THE STACK
2840  003277' 000000  000c21
2841  003300' 001000  000305          41420           PUSH    B
2842  003301' 001000  000063          41440           INX     SP              ;THE "TOKEN" ONLY TAKES ONE BYTE OF
2843                                  41460                                   ;STACK SPACE
2844                                  41480   ;       JMP     NEWSTT          ;ALL DONE
2845                                  41500   PAGE
```

```
                                        41520  SUBTTL NEW STATEMENT FETCHER
                                        41540  ;
                                        41560  ; BACK HERE FOR NEW STATEMENT, CHARACTER POINTED TO BY (H,L)
                                        41580  ; "H" ON END-OF-LINE, THE ADDRESS OF THIS LOCATION IS
                                        41600  ; LEFT ON THE STACK WHEN A STATEMENT IS EXECUTED SO
                                        41620  ; IT CAN MERELY DO A RETURN WHEN IT IS DONE.
                                        41640  ;
2853  003302'                           41660  NEWSTT:
2854                                    41680          IFN     LISTEN,<
2855                                    41700          IFN     LENGTH,<
2856  003302' 001000  000333           41720                  IN      0                       ;CHECK FOR A CHARACTER WITHOUT
2857  003305' 000000  000000
2858                                    41740                                                  ;DOING A "CALL" FOR SPEED
2859                  003303'           41760  CNLOCAL==.-1
2860  003304' 001000  000346           41780                  ANI     IDONE                   ;CHARACTER THERE?
2861  003305' 000000  000001
2862  003306' 001000  000314           41800                  CZ      CNTCON>                 ;SEE IF IT'S CONTROL-C
2863  003307' 000000  003465'
2864  003310' 000000  003273'
2865                                    41820          IFE     LENGTH,<
2866                                    41840                  CALL    ISCNTC>>
2867                                    41860          IFN     LENGTH,<
2868  003311' 001000  000042           41880                  SHLD    TEMP>                   ;USED BY CONTINUE AND INPUT AND CLEAR
2869  003312' 000000  001605'
2870  003313' 000000  003307'
2871                                    41900                                                  ;TO REMEMBER HOW TO RESTART THIS
2872                                    41920                                                  ;STATEMENT
2873                                    41940  IFN     LPTSW,<XRA      A                       ;FORCE PRINT TO GO TO TTY AFTER LPRINT
2874                                    41960          STA     PRTFLG>
2875  003314' 001002  000176           41980          MOV     A,M                     ;GET CURRENT CHARACTER
2876                                    42000                                                  ;WHICH TERMINATED THE LAST STATEMENT
2877  003315' 001002  000576           42020          CPI     ":"                     ;IS IT A COLON?
2878  003316' 000000  000072
2879  003317' 001000  000312           42040          JZ      GONE
2880  003320' 000000  003370'
2881  003321' 000000  003312'
2882  003322' 001000  000267           42060          ORA     A
2883  003323' 001000  000302           42080          JNZ     SNERR                   ;MUST BE A ZERO
2884  003324' 000000  002072'
2885  003325' 000000  003320'
2886  003326' 001000  000043           42100          INX     H
2887  003327' 001000  000176           42120          MOV     A,M                     ;CHECK POINTER TO SEE IF
2888                                    42140                                          ;IT IS ZERO, IF SO WE ARE AT THE
2889                                    42160                                          ;END OF THE PROGRAM
2890  003330' 001000  000043           42180          INX     H
2891  003331' 001000  000266           42200          ORA     M                       ;OR IN HIGH PART
2892  003332' 001000  000043           42220          INX     H
2893  003333' 001000  000312           42240          JZ      ENDCON                  ;RAN OFF THE END -- OK
2894  003334' 000000  003501'
2895  003335' 000000  003424'
2896  003336' 001000  000136           42260          MOV     E,M
2897  003337' 001000  000043           42280          INX     H
2898  003340' 001000  000126           42300          MOV     D,M                     ;GET LINE # IN [D,E]
```



```
2899  003341' 001000  000353           42320          XCHG                            ;[H,L]=LINE #
2900  003342' 001000  000042           42340          SHLD    CUR_IN                  ;SETUP CUR_IN WITH THE CURRENT LINE #
2901  003345' 000000  001687'
2902  003346' 000000  003334'
2903                                    42360  IFE     LENGTH-2,<                      ;TRACE FEATURE
2904  003349' 001000  000072           42380          LDA     TRCFLG                  ;SEE IF TRACE IS ON
2905  003346' 000000  001631'
2906  003347' 000000  003343'
2907  003350' 001000  000267           42400          ORA     A                       ;NON-ZERO MEANS YES
2908  003351' 001000  000312           42420          JZ      NOTTRC                  ;SKIP THIS PRINTING
2909  003352' 000000  003367'
2910  003355' 000000  003346'
2911  003354' 001000  000325           42440          PUSH    D                       ;SAVE THE TEXT POINTER
2912  003355' 001000  000076           42460          MVI     A,"["                   ;FORMAT THE LINE NUMBER
2913  003356' 000000  000133
2914  003357' 001000  000537           42480          OUTCHR                          ;OUTPUT IT
2915  003360' 001000  003315           42500          CALL    LINPRT                  ;PRINT THE LINE # IN [H,L]
2916  003361' 000000  000000*
2917  003362' 000000  003352'
2918  003365' 001000  000076           42520          MVI     A,"]"                   ;SOME MORE FORMATING
2919  003364' 000000  000135
2920  003365' 001000  000337           42540          OUTCHR
2921  003366' 001000  000321           42560          POP     D                       ;[D,E]=TEXT POINTER
2922  003367'                           42580  NOTTRC1:
2923  003367' 001000  000353           42600          XCHG                            ;RESTORE THE TEXT POINTER
2924  003370' 001000  000327           42620  GONE:   CHRGET                          ;GET THE STATEMENT TYPE
2925  003371' 001000  000021           42640          LXI     D,NEWSTT                ;PUSH ON A RETURN ADDRESS OF NEWSTT
2926  003372' 000000  003302'
2927  003373' 000000  003361'
2928  003374' 001000  000325           42660          PUSH    D                       ;STATEMENT
2929  003375' 001000  000310           42680  GONE3:  RZ                              ;IF A TERMINATOR TRY AGAIN
2930                                    42700                                          ;"IF" COMES HERE
2931  003376' 001000  000326           42720  GONE2:  SUI     ENDTK                   ;"ON ... GOTO" AND "ON ... GOSUB" COME HERE
2932  003377' 000000  000000
2933  003400' 001000  000332           42740          JC      LET                     ;MUST BE A LET
2934  003401' 000000  004131'
2935  003402' 000000  003372'
2936                                    42760  NUMCMD==CMATK-ENDTK+1
2937  003403' 001000  000376           42780          CPI     NUMCMD
2938  003404' 000000  000000
2939  003405' 001000  000322           42800          JNC     SNERR                   ;SOME RESERVED WORD,BUT NOT
2940  003406' 000000  002072'
2941  003407' 000000  003401'
2942                                    42820                                          ;A STATEMENT RESERVED WORD
2943  003410' 001000  000007           42840          RLC                             ;MULTIPLY BY 2
2944  003411' 001000  000117           42860          MOV     C,A
2945  003412' 001000  000006           42880          MVI     B,0
2946  003413' 000000  000000
2947  003414' 001000  000353           42900          XCHG
2948  003415' 001000  000041           42920          LXI     H,STMDSP                ;STATEMENT DISPATCH TABLE
2949  003416' 000000  000564'
2950  003417' 000000  003446'
2951  003420' 001000  000011           42940          DAD     B                       ;ADD ON OFFSET
```

```
BASIC MOS 8080  GATES/ALLEN/DAVIDOFF    MACRO 4/(113) 03:12 10-SEP-75 PAGE 11-2
F3      MAC     6-SEP-64 03:11          NEW STATEMENT FETCHER

                                        43960           MOV     C,M             ;PUSH THE ADDRESS TO GO TO ONTO
                                        43980           INX     H               ;THE STACK
                                        44000           MOV     B,M             ;PUSHM SAVES BYTES BUT NOT SPEED
                                        44020           PUSH    B
                                        44040           XCHG                    ;RESTORE THE TEXT POINTER
                                        44060   IFE     LENGTH,<
                                        44080           CHRGET                  ;EAT THE FIRST CHARACTER
                                        44100           RET>                    ;GO DO THE STATEMENT
                                        44120   IFN     LENGTH,<
                                        44140   CHRGTI  INX     H               ;DUPLICATION OF CHRGET RST FOR SPEED
                                        44160           MOV     A,M             ;SEE CHRGET RST FOR EXPLANATION
                                        44180           CPI     ":"
                                        44200           RNC>
                                        44220   ;
                                        44240   ; CHRCON IS THE CONTINUATION OF THE CHRGET RST
                                        44260   ;
                                        44280   CHRCONI CPI     " "             ;MUST SKIP SPACES
                                        44300           JZ      CHRGTK          ;GET ANOTHER CHARACTER
                                        44320           CPI     "0"             ;ALL CHARACTERS GREATER THAN
                                        44340                                   ;"9" HAVE RETURNED, SO SEE IF NUMERIC
                                        44360           CMC                     ;MAKE NUMERICS HAVE CARRY ON
                                        44380           INR     A               ;SET ZERO IF (A)=0
                                        44400           DCR     A
                                        44420           RET
                                        44440   PAGE


BASIC MOS 8080  GATES/ALLEN/DAVIDOFF    MACRO 4/(113) 03:12 10-SEP-75 PAGE 12
F3      MAC     6-SEP-64 03:11          RESTORE,STOP,END,LINGET,CHRCON

                                        44460   SUBTTL RESTORE,STOP,END,LINGET,CHRCON
                                        44480
                                        44500   RESTOR: XCHG                    ;SAVE (H,L) IN (D,E)
                                        44520           LHLD    TXTTAB
                                        44540           DCX     H               ;INITIALIZE DATPTR TO (TXTTAB)-1
                                        44560   RESFIN: SHLD    DATPTR          ;READ FINISHES COME TO RESFIN
                                        44580           XCHG                    ;GET THE TEXT POINTER BACK
                                        44600           RET
                                        44620
                                        44640   IFN     LISTEN,<
                                        44660   ISCNTC: IN      0
                                        44680           CNLEAZ=.-1              ;CONSOLE COMMAND CHANGE LOC
                                        44700           ANI     IDONE
                                        44720           RNZ                     ;IF NO CHARACTERS THEN NO ^C
                                        44740   CNTLCN: CALL    INCHR
                                        44760           CPI     3               ;STOP CHARACTER IS ^C
                                        44780   IFE     LENGTH,<
                                        44800           JMP     STOP>>
                                        44820   IFN     LENGTH,<
                                        44840   STOP:   RNZ                     ;RETURN IF NOT CONTROL-C AND MAKE
                                        44860                                   ;SURE "STOP" STATEMENTS HAVE A TERMINATOR
                                        44880           XRD     "D1000,"D366    ;SETUP (A) AS A FLAG WHETHER
                                        44900                                   ;TO TYPE THE BREAK MESSAGE
                                        44920   END:    RNZ                     ;MAKE SURE "END" STATEMENTS HAVE A TERMINATOR
                                        44940           SHLD    TEMP            ;SAVE FOR "CONTINUE"
                                        44960   STPEND: POP     B               ;POP OFF NEWSTT ADDRESS
                                        44980   ENDCON: PUSH    PSW             ;SAVE THE MESSAGE FLAG
                                        45000                                   ;ZERO MEANS DON'T PRINT "BREAK"
                                        45020           LHLD    CURLIN          ;SAVE CURLIN
                                        45040           MOV     A,L
                                        45060           ANA     H               ;SEE IF IT WAS DIRECT
                                        45080           INR     A
                                        45100           JZ      DIRIS           ;IF NOT SET UP FOR CONTINUE
                                        45120           SHLD    OLDLIN          ;SAVE OLD LINE #
                                        45140           LHLD    TEMP            ;GET POINTER TO START OF STATEMENT
```

```
                                                    ;
                                                    ;
                                                    ;
```

```
                                                45680   FCERR:  MVI     E,ERRFC         ;TOO BIG, FUNCTION CALL ERROR
                                                45700           JMP     ERROR>
                                                45720   IFE     LENGTH-2,<
003626' 001040  000527          003626' 001040  45740   INTIDX: CHRGET
003629' 001000  000319          003629' 001000  45760   INT.DE: CALL    FMEVL           ;EVALUATE A FORMULA
003624' 000000  003536'
003625' 000000  003607'
003626' 001040  000545          003626' 001040  45780           PUSH    H               ;SAVE THE TEXT POINTER
003627' 001000  000515          003627' 001000  45800           CALL    FMINT           ;CONVERT THE FAC TO AN INTEGER
003630' 000000  000666*
003631' 000000  003624'
003632' 001040  000174          003632' 001040  45820           MOV     A,M             ;SEE IF THE RESULT IS NEGATIVE
003633' 001000  000267          003633' 001000  45840           ORA     A               ;BY LOOKING AT (M)'S MSB
003634' 001000  000312          003634' 001000  45860           JM      FCERR           ;DON'T ALLOW NEGATIVE NUMBERS
003635' 000000  010776'
003636' 000000  003630'
003637' 001000  000553          003637' 001000  45880           XCHG                    ;RETURN THE INTEGER IN (D,E)
003640' 001000  000341          003640' 001000  45900           POP     H               ;RESTORE THE TEXT POINTER
003641' 001000  000311          003641' 001000  45920           RET>
                                                45940
                                                45960   ;
                                                45980   ; L.INGET READS A LINE # FROM THE CURRENT TEXT POSITION
                                                46000   ;
                                                46020   ; LINE NUMBERS RANGE FROM 0 TO 65529
                                                46040   ;
                                                46060   ; (D,E) IS SMASHED.
                                                46080   ;
                                                46100   ; ANSWER RETURNED IN (D,E),
                                                46120   ; (H,L) IS UPDATED TO POINT TO THE TERMINATING CHARACTER
                                                46140   ; AND (A) CONTAINS THE TERMINATING CHARACTER WITH CONDITION
                                                46160   ; CODES SET UP TO REFLECT ITS VALUE.
                                                46180   ;
003642' 001040  000533          003642' 001040  46200   LINGET: DCX     H
003643' 001000  000021          003643' 001000  46220   LINGT2: LXI     D,$CODE         ;ZERO ACCUMULATED LINE #
003644' 000000  000000'
003645' 000000  003635'
003646' 001000  000527          003646' 001000  46240   MOR.IN: CHRGET
003647' 001000  001320          003647' 001000  46260           RNC                     ;WAS IT A DIGIT
003650' 001000  000345          003650' 001000  46280           PUSH    H
003651' 001000  000365          003651' 001000  46300           PUSH    PSW
003652' 001000  000041          003652' 001000  46320           LXI     H,$CODE+6552    ;SEE IF THE LINE # IS TOO BIG
003653' 000000  014630'
003654' 000000  003644'
003655' 001000  000347          003655' 001000  46340           COMPAR
003656' 001000  000332          003656' 001000  46360           JC      SNERR           ;YES, SYNTAX ERROR
003657' 000000  002072'
003660' 000000  003653'
003661' 001000  000142          003661' 001000  46380           MOV     H,D             ;SAVE (D,E)
003662' 001000  000153          003662' 001000  46400           MOV     L,E
003663' 001000  000031          003663' 001000  46420           DAD     D
003664' 001000  000051          003664' 001000  46440           DAD     H
003665' 001000  000031          003665' 001000  46460           DAD     D
003666' 001000  000051          003666' 001000  46480           DAD     H               ;PUTTING (D,E)*10 INTO (H,L)
```



```
003667' 001000  000361          003667' 001000  46500           POP     PSW
003670' 001000  000326          003670' 001000  46520           SUI     '0'
003671' 000000  000060
003672' 001000  000137          003672' 001000  46540           MOV     E,A
003673' 001000  000026          003673' 001000  46560           MVI     D,0
003674' 000000  000000
003675' 001000  000031          003675' 001000  46580           DAD     D               ;ADD THE NEW DIGIT
003676' 001000  000353          003676' 001000  46600           XCHG
003677' 001000  000341          003677' 001000  46620           POP     H               ;GET BACK TEXT POINTER
003700' 001000  000303          003700' 001000  46640           JMP     MORLIN
003701' 000000  003646'
003702' 000000  003657'
                                                46660   IFN     LENGTH,<
003703' 001000  000312          003703' 001000  46680   CLEAR:  JZ      CLEARC          ;IF NO FORMULA JUST CLEAR
003704' 000000  002445'
003705' 000000  003703'
003706' 001000  000315          003706' 001000  46700           CALL    INTID2          ;GET AN INTEGER INTO (D,E)
003707' 000000  003623'
003710' 000000  003704'
003711' 001000  000053          003711' 001000  46720           DCX     H
003712' 001000  000327          003712' 001000  46740           CHRGET                  ;SEE IF ITS THE END
003713' 001000  000300          003713' 001000  46760           RNZ                     ;SHOULD FINISH THERE
003714' 001000  000345          003714' 001000  46780           PUSH    H               ;SAVE TXTPTR
003715' 001000  000052          003715' 001000  46800           LHLD    MEMSIZ          ;GET HIGHEST ADDRESS
003716' 000000  001545'
003717' 000000  003707'
003720' 001000  000175          003720' 001000  46820           MOV     A,L             ;SUBTRACT (H,L)-(D,E) INTO (D,E)
003721' 001000  000223          003721' 001000  46840           SUB     E
003722' 001000  000137          003722' 001000  46860           MOV     E,A
003723' 001000  000174          003723' 001000  46880           MOV     A,H
003724' 001000  000032          003724' 001000  46900           SBB     D
003725' 001000  000127          003725' 001000  46920           MOV     D,A
003726' 001000  000332          003726' 001000  46940           JC      SNERR           ;WANTED MORE THAN TOTAL!
003727' 000000  002072'
003730' 000000  003716'
003731' 001000  000052          003731' 001000  46960           LHLD    VARTAB          ;TOP LOCATION IN USE
003732' 000000  001621'
003733' 000000  003727'
003734' 001000  000001          003734' 001000  46980           LXI     B,$CODE+48      ;LEAVE BREATHING ROOM
003735' 000000  000060'
003736' 000000  003732'
003737' 001000  000011          003737' 001000  47000           DAD     B
003740' 001000  000347          003740' 001000  47020           COMPAR                  ;ROOM?
003741' 001000  000322          003741' 001000  47040           JNC     OMERR           ;NO, DON'T EVEN CLEAR
003742' 000000  002057'
003743' 000000  003735'
003744' 001000  000353          003744' 001000  47060           XCHG                    ;NEW STACK LOCATION (H,L)
003745' 001000  000042          003745' 001000  47080           SHLD    STKTOP          ;SET UP NEW STACK LOCATION
003746' 000000  001615'
003747' 000000  003742'
003750' 001000  000341          003750' 001000  47100           POP     H               ;REGAIN THE TEXT POINTER
003751' 001000  000305          003751' 001000  47120           JMP     CLEARC>         ;GO CLEAR
003752' 000000  002443'
```

```
 3247  003753' 000000  003746'
 3248                          47140   PAGE
```



```
 3249                          47160   SUBTTL  RUN,GOTO,GOSUB,RETURN
 3250                          47180   IFN     LENGTH,<
 3251  003754' 001000  000512  47200   RUN:    JZ      RUNC            ;NO LINE # ARGUMENT
 3252  003755' 000000  002437'
 3253  003756' 000000  003152'
 3254                          47220                                   ;CLEAN UP,SET (H,L)=(TXTTAB)-1 AND
 3255                          47240                                   ;RETURN TO NEWSTT
 3256  003757' 001000  000315  47260           CALL    CLEARC          ;CLEAN UP -- RESET THE STACK
 3257  003760' 000000  002443'
 3258  003761' 000000  003755'
 3259                          47280                                   ;DATPTR,VARIABLES ...
 3260                          47300                                   ;(H,L) IS THE ONLY THING PRESERVED
 3261  003762' 001000  000001  47320           LXI     B,NEWSTT
 3262  003763' 000000  003302'
 3263  003764' 000000  003760'
 3264  003765' 001000  000303  47340           JMP     RUNC2>          ;PUT "NEWSTT" ON AND FALL INTO "GOTO"
 3265  003766' 000000  004007'
 3266  003767' 000000  003763'
 3267                          47360   ;
 3268                          47380   ; A GOSUB ENTRY ON THE STACK HAS THE FOLLOWING FORMAT
 3269                          47400   ;
 3270                          47420   ; LOW ADDRESS
 3271                          47440   ;
 3272                          47460   ;       A TOKEN EQUAL TO GOSUTK 1 BYTE
 3273                          47480   ;       THE LINE # OF THE THE GOSUB STATEMENT 2 BYTES
 3274                          47500   ;       A POINTER INTO THE TEXT OF THE GOSUB 2 BYTES
 3275                          47520   ;
 3276                          47540   ; HIGH ADDRESS
 3277                          47560   ;
 3278                          47580   ; TOTAL 5 BYTES
 3279                          47600   ;
 3280  003770' 001000  000315  47620   GOSUB:  CALL    GETSTK          ;MAKE SURE THERE IS ROOM
 3281  003771' 000000  002624'
 3282  003772' 000000  003766'
 3283  003773' 000000  000003  47640           3
 3284  003774' 001000  000501  47660           POP     B               ;POP OFF RETURN ADDRESS OF "NEWSTT"
 3285  003775' 001000  000545  47680           PUSH    H               ;REALLY PUSH THE TEXT POINTER
 3286  003776' 001000  000345  47700           PUSH    H               ;SAVE TEXT POINTER
 3287  003777' 001000  000052  47720           LHLD    CURLIN          ;GET THE CURRENT LINE #
 3288  004000' 000000  001607'
 3289  004001' 000000  003771'
 3290  004002' 001000  000343  47740           XTHL                    ;PUT CURLIN ON THE STACK AND (H,L)=TEXT PTR
 3291  004003' 001000  000026  47760           MVI     D,GOSUTK        ;LEAVE A GOSUB TOKEN
 3292  004004' 000000  000C,4
 3293                          47780                                   ;ON THE STACK
 3294  004005' 001000  000325  47800           PUSH    D
 3295  004006' 001000  000043  47820           INX     SP              ;THE GOSUB TOKEN TAKES ONLY ONE BYTE
 3296  004007' 001000  000305  47840   RUNC2:  PUSH    B               ;RESTORE RETURN ADDRESS
 3297                          47860                                   ;OF "NEWSTT"
 3298                          47880   ;
 3299                          47900   ; IN THE 4K VERSION WE START AT THE BEGINNING
 3300                          47920   ; AND SEARCH, IN THE 8K WE START WHERE WE
 3301                          47940   ; ARE IF WE ARE  GOING TO A FORWARD LOCATION.
```

```
                                47480   ;
3303  004010' 001000  000315    47980   GOTO1:  CALL    LINGET          ;PICK UP THE LINE #
3304  004013' 000000  005642'
3305  004016' 000000  004006'
3306                            48000                                   ;AND PUT IT IN (D,E)
3307                            48020           IFE     LENGTH,<RNZ     ;SHOULD END WITH A LINE
3308                            48040                                   ;TERMINATOR -- BLOW HIM UP
3309                            48060                                   ;IF IT DOESN'T
3310                            48080                                   ;(ON GOTO MAKES THIS WRONG
3311                            48100                                   ;IN OTHER VERSONS)
3312                            48120           CALL    FNDLIN>
3313                            48140   IFN     LENGTH,<
3314  004015' 001000  000315    48160           CALL    REM             ;SKIP TO THE END OF THIS LINE
3315  004018' 000000  004074'
3316  004019' 000000  004011'
3317  004016' 001000  000345    48180           PUSH    H               ;SAVE THE POINTER
3318  004017' 001000  000052    48200           LHLD    CURLIN          ;GET THE CURRENT LINE #
3319  004020' 000000  001607'
3320  004021' 000000  004014'
3321  004022' 001000  000347    48220           COMPAR                  ;(D,E) CONTAINS WHERE WE ARE GOING
3322                            48240                                   ;(M,L) CONTAINS THE CURRENT LINE#
3323                            48260                                   ;SO COMPARING THEM TELL US WHETHER TO
3324                            48280                                   ;START SEARCHING FROM WHERE WE ARE OR
3325                            48300                                   ;TO START SEARCHING FROM THE BEGINNING
3326                            48320                                   ;OF TXTTAB
3327  004023' 001000  000341    48340           POP     H               ;(M,L)=CURRENT POINTER
3328  004024' 001000  000043    48360           INX     H               ;POINT AT THE LINK BEYOND IT
3329  004025' 001000  000334    48380           CC      LOOP            ;SEARCH FROM THIS POINT
3330  004026' 000000  004374'
3331  004027' 000000  004020'
3332  004030' 001000  000324    48400           CNC     FNDLIN>         ;SEARCH FROM THE BEGINNING -- ACTUALLY
3333  004031' 000000  004571'
3334  004032' 000000  004026'
3335                            48420                                   ;SEARCH AGAIN IF ABOVE SEARCH FAILED
3336  004033' 001000  000140    48440           MOV     M,B
3337  004034' 001000  000151    48460           MOV     L,C
3338  004035' 001000  000053    48480           DCX     H
3339  004036' 001000  000330    48500           RC                      ;IF A MATCH WE ARE DONE
3340  004037' 001000  000036    48520   USERR:  MVI     E,ERRUS
3341  004040' 000000  000010
3342  004041' 001000  000303    48540           JMP     ERROR           ;ERRMATCH,SO IF NO MATCH WE
3343  004042' 000000  00210d'
3344  004043' 000000  004011'
3345                            48560                                   ;GIVE A "US" ERROR
3346                            48580   ;
3347                            48600   ; SEE "GOSUB" FOR THE FORMAT OF THE STACK ENTRY
3348                            48620   ; "RETURN" RESTORES THE LINE NUMBER AND TEXT POINTER ON THE STACK
3349                            48640   ; AFTER ELIMINATING ALL THE "FOR" ENTRIES IN FRONT OF THE "GOSUB"
3350                            48660   ; ENTRY
3351                            48680   ;
3352  004044' 001000  000300    48700   RETURN: RNZ                     ;BLOW HIM UP IF THERE ISN'T A TERMINATOR
3353  004045' 001000  000026    48720           MVI     D,255           ;MAKE SURE THIS VARIABLE POINTER
3354  004046' 000000  000377
```



```
                                48740                                   ;IN (D,E) NEVER GETS MATCHED
3356  004047' 001000  000315    48760           CALL    FNDFOR          ;GO PAST ALL THE "FOR" ENTRIES
3357  004050' 000000  001744'
3358  004051' 000000  004042'
3359  004052' 001000  000371    48780           SPHL                    ;UPDATE THE STACK
3360  004053' 001000  000376    48800           CPI     GOSUTK
3361  004054' 000000  000214
3362  004055' 001000  000036    48820           MVI     E,ERNRG         ;ERROR ERNRG IS "RETURN WITHOUT GOSUB"
3363  004056' 000000  000005
3364  004057' 001000  000302    48840           JNZ     ERROR
3365  004060' 000000  00210d'
3366  004061' 000000  004050'
3367  004062' 001000  000341    48860           POP     H               ;GET LINE # "GOSUB" WAS FROM
3368  004063' 001000  000042    48880           SHLD    CURLIN          ;PUT IT INTO CURLIN
3369  004064' 000000  001b07'
3370  004065' 000000  004064'
3371  004066' 001000  000041    48900           LXI     H,NEWSTT
3372  004067' 000000  003302'
3373  004070' 000000  004064'
3374  004071' 001000  000345    48920           XTHL                    ;PUT RETURN ADDRESS OF "NEWSTT"
3375                            48940                                   ;BACK ONTO THE STACK, GET TEXT POINTER
3376                            48960                                   ;FROM "GOSUB"
3377                            48980                                   ;SKIP OVER SOME CHARACTERS
3378                            49000                                   ;SINCE WHEN "GOSUB" STUCK THE TEXT POINTER
3379                            49020                                   ;ONTO THE STACK THE LINE # ARGUMENT HADN'T
3380                            49040                                   ;BEEN READ IN YET.
3381
3382
3383                            49100   IFN     STRING,<
3384  004072' 001000  000001    49120   DATA:   XWD     "01000,"0       ;"LXI B," TO PICK UP ":" INTO C AND SKIP
3385  004073' 000000  000072    49140           ":"                     ;"DATA" TERMINATES ON ":"
3386                            49160                                   ;AND 0, ":" ONLY APPLIES IF
3387                            49180                                   ;QUOTES HAVE "ATCHED UP
3388  004074'                   49200   IFE     LENGTH-2,<ELSE1>        ;EXECUTED "ELSE"S ARE SKIPPED
3389                            49220   >
3390                            49240   ; NOTE: REM MUST PRESERVE (D,E) BECAUSE OF "GO TO" AND ERROR
3391                            49260   ;
3392  004074' 001000  000016    49280   REM:    XWD     "01000,"016     ;MVI C,   THE ONLY TERMINATOR IS 0
3393  004075' 001000  000000    49300           XWD     "01000,0        ;NO-OPERATION
3394                            49320                                   ;"DATA" ACTUALLY EXECUTES THIS 0
3395  004076' 001000  000006    49340           MVI     B,0             ;INSIDE QUOTES THE ONLY TERMINATOR IS ZERO
3396  004077' 000000  000000
3397  004100' 001000  000171    49360   EXCHQT: MOV     A,C             ;WHEN A QUOTE IS SEEN THE SECOND
3398  004101' 001000  002110    49380           MOV     C,B             ;TERMINATOR IS TRADED, SO IN "DATA"
3399  004102' 001000  000107    49400           MOV     B,A             ;COLONS INSIDE QUOTATIONS WILL HAVE NO EFFECT
3400  004103' 001000  000176    49420   REMER:  MOV     A,M             ;GET THE CHARACTER
3401  004104' 001000  000267    49440           ORA     A               ;ZERO IS ALWAYS A TERMINATOR
3402  004105' 001000  000510    49460           RZ
3403  004106' 001000  000270    49480           CMP     B               ;TEST FOR THE OTHER TERMINATOR
3404  004107' 001000  000510    49500           RZ
3405  004110' 001000  000043    49520           INX     H
3406  004111' 001000  000376    49540           CPI     34              ;IS IT A QUOTE?
3407  004112' 000000  000042
```

```
3408  004115' 001000  000512          49560          JZ      EXCHOT          ;IF SO TIME TO TRADE
3409  004116' 000000  004100'
3410  004117' 000000  004667'
3411                                   49580  IFE     LENGTH-2,<
3412                                   49600  ;
3413                                   49620  ; WHEN AN "IF" TAKES A FALSE BRANCH IT MUST FIND THE APPROPRIATE "ELSE"
3414                                   49640  ; TO START EXECUTION AT.  "DATA" COUNTS THE NUMBER OF "IF"S
3415                                   49660  ; IT SEES SO THAT THE "ELSE" CODE CAN MATCH "ELSE"S WITH
3416                                   49680  ; "IF"S, THE COUNT IS KEPT IN [L]
3417                                   49700  ;
3418  004118' 001000  000326          49720          SUI     IFTK            ;IS IT AN "IF"
3419  004119' 000000  000D1C
3420  004120' 001000  000382          49740          JNZ     REMER           ;IF NOT, CONTINUE ON
3421  004121' 000000  004104'
3422  004122' 000000  004114'
3423  004123' 001000  000270          49760          CMP     B               ;SINCE "REM" CAN'T SMASH
3424                                   49780                                  ;[D,E] WE HAVE TO BE CAREFUL
3425                                   49800                                  ;SO ONLY IF B DOESN'T EQUAL
3426                                   49820                                  ;ZERO WE INCREMENT D, (THE "IF" COUNT)
3427  004124' 001000  000D12          49840          AOC     D
3428  004125' 001000  000127          49860          MOV     D,A>            ;UPDATE [D]
3429  004126' 001000  000303          49880          JMP     REMER>
3430  004127' 000000  004105'
3431  004130' 000000  004121'
3432                                   49900  ;
3433                                   49902  ; WITHOUT STRINGS THERE IS NO NEED TO WATCH QUOTATIONS
3434                                   49904  ; SO [B] IS SET UP AS THE SECONDARY TERMINATOR
3435                                   49906  ; AND A SCAN IS MADE FOR [B] OR ZERO
3436                                   49908  ;
3437                                   49920  IFE     STRING,<
3438                                   49940  DATA:   XWD     "D1000,1        ;MAKE [C]="I" AND SKIP
3439                                   49960          "I"
3440                                   49980  REM:    XWD     "D1000,16       ;MAKE [C]=0
3441                                   50000          XWD     "D1000,0        ;ZERO AND NO-OPERATION
3442                                   50020  LOOPDR: MOV     A,M
3443                                   50040          ORA     A               ;ALWAYS STOP ON ZERO
3444                                   50060          RZ
3445                                   50080          CMP     C               ;CHECK FOR "I" IN DATA
3446                                   50100          RZ
3447                                   50120          INX     H               ;LOOK AT NEXT CHARACTER
3448                                   50140          JMP     LOOPDR>
3449                                   50160  PAGE
```



```
3450                                   50180  SUBTTL "LET"
3451  004131' 001000  000315          50200  LET:    CALL    PTRGET          ;GET THE POINTER TO THE VARIABLE
3452  004132' 000000  006525'
3453  004133' 000000  004127'
3454                                   50220                                  ;NAMED IN TEXT AND PUT
3455                                   50240                                  ;IT ON THE STACK, [H,L] REMAINS
3456                                   50260                                  ;THE TEXT POINTER AND A,PSW ARE SET-P
3457                                   50280                                  ;AS THE TERMINATING CHARACTER.
3458  004134' 001000  000317          50300          SYNCHK  EQULTK          ;CHECK FOR "="
3459  004135' 000000  004260
3460  004136'                        50320  REDINP:
3461                                   50340  IFN     STRING,<
3462  004136' 001000  000D7E          50360          LDA     VALTYP
3463  004137' 000000  001543'
3464  004140' 000000  004132'
3465  004141' 001000  000365          50380          PUSH    PSW>
3466  004142' 001000  000325          50400          PUSH    D
3467  004143' 001000  000315          50420          CALL    FRMEVL          ;GET THE VALUE OF THE FORMULA
3468  004144' 000000  005336'
3469  004145' 000000  004137'
3470                                   50440                                  ;INTO FAC
3471  004146' 001000  000343          50460          XTHL                    ;[H,L]=POINTER TO VARIABLE
3472                                   50480                                  ;TEXT POINTER TO ON TOP OF STACK
3473  004147' 001000  000042          50500          SHLD    TEMP            ;SAVE VARIABLE POINTER FOR "FOR"
3474  004150' 000000  001D3F'
3475  004151' 000000  004144'
3476                                   50520  IFN     STRING,<
3477  004152' 001000  000321          50540          POP     D               ;GET TEXT POINTER OFF
3478  004153' 001000  000361          50560          POP     PSW             ;GET THE VALTYP OF THE
3479                                   50580                                  ;VARIABLE INTO [A]
3480  004154' 001000  000325          50581          PUSH    D               ;RESAVE THE TEXT-POINTER
3481                                   50582  IFE     LENGTH-2,<
3482  004155' 001000  000376          50584          CPI     3               ;SEE IF IT'S A STRING
3483  004156' 000000  000005
3484  004157' 001000  000345          50586  INPCOM: PUSH    H               ;SAVE THE POINTER AT THE VALUE POSITION
3485  004160' 001000  000302          50588          JNZ     COPNUM          ;NUMERIC, SO FORCE IT AND COPY
3486  004161' 000000  004F3F'
3487  004162' 000000  004156'
3488  004163' 001000  000315          50590          CALL    CHKSTR>         ;MAKE SURE THE FORMULA WAS A STRING
3489  004164' 000000  000000
3490  004165' 000000  004161'
3491                                   50600  IFN     LENGTH-2,<
3492                                   50620          RAR                     ;CARRY SET FOR STRING OFF
3493                                   50640                                  ;FOR NUMERIC
3494                                   50660          CALL    CHKVAL          ;MAKE SURE VALTYP MATCHES CARRY
3495                                   50680                                  ;AND SET THE ZERO FLAG
3496                                   50700                                  ;ON A NUMERIC VALTYP
3497                                   50860          JZ      COPNUM          ;IF A NUMBER  COPY
3498                                   50880  INPCOM: PUSH    H>              ;SAVE POINTER AT VARIABLE
3499  004166' 001000  000352          50900          LHLD    FACLO           ;GET POINTER TO THE DESCRIPTOR OF THE RESULT
3500  004167' 000000  001D37'
3501  004170' 000000  004164'
3502  004171' 001000  000345          50920          PUSH    H               ;SAVE THE POINTER AT THE DESCRIPTOR
```

```
                                                    ;SEE IF IT POINTS INTO STRING SPACE
                                                    ;IF NOT DON'T COPY
                                                    ;GET BACK THE POINTER AT THE DESCRIPTOR
                                                    ;DON'T COPY LITERALS
                                                    ;NOW, SEE IF ITS A VARIABLE
                                                    ;BY SEEING IF THE  DESCRIPTOR
                                                    ;IS BEYOND [VARTAB], IF SO COPY
                                                    ;GET THE LENGTH AND SAVE IT
                                                    ;SINCE WE ARE GOING TO SET IT TO
                                                    ;0 SO FRETMP DOESN'T UPDATE FRETOP
                                                    ;SET IT TO 0 -- ELIMINATING NULL
                                                    ;STRING IN FRETMP IS HARMLESS
                                                    ;PUT 0 IN THE LENGTH FIELD
                                                    ;FREE IT UP
                                                    ;GET LENGTH BACK
                                                    ;REPLACE IT ( FRETMP RETURNS [D,E] IN [H,L] )
                                                    ;PUT THE DESCRIPTOR POINTER BACK IN [D,E]
                                                    ;GET THE PLACE OF THE NEW VARIABLE
                                                    ;COPY THE DESCRIPTOR
                                                    ;FORM LLA TYPE TO CONFORM
                                                    ;TO THE VARIABLE ITS BEING ASSIGNED TO
                                                    ;TABLE OF FORCE ROUTINES
                                                    ;[B,C]=TWO BYTE OFFSET
```



```
                                                    ;[H,L]=ADDRESS TO GO TO
                                                    ;RETURN TO PUTVAL
                                                    ;DISPATCH TO FORCE
                                                    ;GET THE POINTER OF WHERE TO STORE
                                                    ;THE VALUE
                                                    ;SAVE IT BACK
                                                    ;MOVE THE VALUE IN
                                                    ;SAVE THE VARIABLE POINTER FOR "FOR"
                                                    ;TRANSFER THE VALUE
                                                    ;"FOR" WANTS VARIABLE POINTER IN
                                                    ;[D,E] FOR ENDFOR
                                                    ;GET THE TEXT POINTER
```

```
                                52100   SUBTTL ON GOTO CODE
                                52120   IFN     LENGTH,<
3584  004271' 001600  002515    52160   ONGOTO: CALL    GETBYT          ;GET VALUE INTO [E]
3585  004274' 0C4000  011020'
3586  004275' 000000  004264'
3587  004276' 001600  002176    52180           MOV     A,M             ;GET THE TERMINATOR BACK
3588  004275' 001700  000107    52200           MOV     B,A             ;SAVE THIS CHARACTER FOR LATER
3589  004276' 001600  002576    52220           CPI     GOSUB           ;AN "ON ... GOSUB" PERHAPS?
3590  004277' 000000  000214    52240           JZ      ISGOSU          ;YES, SOME FEATURE USE
3591  004300' 001700  002512    52260           SYNCHK  GOTOTK          ;OTHERWISE MUST BE "GOTO"
3592  004301' 000000  004302'
3593  004302' 000000  002712'
3594  004303' 001000  002517    52280           DCX     H               ;BACK UP CHARACTER POINTER
3595  004304' 000000  010210    52300   ISGOSU: MOV     C,E             ;GET COUNT INTO [C]
3596  004305' 001700  000115    52320   LOOPON: DCR     C               ;SEE IF ENOUGH SKIPS
3597  004306' 001000  000017    52340           MOV     A,B             ;PUT DISPATCH CHARACTER IN PLACE
3598  004311' 000000  002516    52360           JZ      GOME2           ;IF DONE, GO OFF
3599  004312' 000000  003376'
3600  004313' 001000  000515    52380           CALL    LINGT2          ;SKIP OVER A LINE #
3601  004314' 000000  003643'
3602  004315' 000000  004312'
3603  004316' 001000  000376    52400           CPI     44              ;A COMMA
3604  004317' 001600  002576    52420           RNZ                     ;IF A COMMA DOESN'T DELIMIT THE END OF
3605  004320' 000000  000054    52440                                   ;THE LAST LINE # MUST BE THE END OF THE LINE
3606  004321' 001000  000303    52460           JMP     LOOPON>         ;CONTINUE GOBBLING LINE #S
3607  004322' 001000  002503
3608  004325' 000000  004307'
3609  004326' 000000  004315'
3610
                                52500   PAGE
```



```
                                52520   SUBTTL IF ... THEN CODE
3616  004323' 001000  002515    52540   IFE     CALL    FRMEVL          ;EVALUATE A FORMULA
3617  004326' 000000  003536'
3618  004327' 000000  004325'
3619                            52560   IFE     LENGTH,<
3620                            52580   IFN     STRING,<
3621                            52600           LDA     VALTYP          ;GET VALUE TYPE INTO [A]
3622                            52620           PUSH    PSW>>           ;SAVE THE VALUE TYPE ON THE STACK
3623  004330' 001700  002176    52640           MOV     A,M             ;GET TERMINATING CHARACTER OF FORMULA
3624                            52660   IFE     LENGTH,<
3625                            52680           CALL    PUSHF           ;ONTO THE STACK
3626                            52700           MVI     D,0             ;KEEPS RELATIONAL OPERATOR MEMORIES
3627                            52720                                   ;LESS THAN #4
3628                            52740                                   ;EQUAL     #2
3629                            52760                                   ;GREATER THAN #1
3630                            52780   LOOPIF: BUI     GREATK          ;CHECK FOR A RELATIONAL OPERATOR
3631                            52800           JC      ENDREL          ;NOPE
3632                            52820   NUMREL=.-LSSTK-GREATK+1         ;NUMBER OF RELATIONAL OPERATORS
3633                            52840           CPI     NUMREL          ;IS THIS ONE OF THEM?
3634                            52860           JNC     ENDREL          ;NO SEE WHAT WE HAVE
3635                            52880           CPI     1               ;SETUP BITS BY MAPPING
3636                            52900           RAL                     ;0 TO 1, 1 TO 2 AND 2 TO 4
3637                            52920           ORA     D               ;OR WITH EARLIER BITS
3638                            52940           MOV     D,A             ;STORE NEW BITS
3639                            52960           CHRGET                  ;GET NEW CHARACTER
3640                            52980           JMP     LOOPIF          ;SEE IF RELATIONAL
3641                            53000   ENDREL: MOV     A,D             ;GET RELATIONAL MEMORIES
3642                            53020           ORA     A               ;SEE IF THERE ARE ANY
3643                            53040           JZ      SNERR           ;NO RELATIONAL OPERATORS!
3644                            53060           PUSH    PSW             ;SAVE RELATIONAL MEMORIES
3645                            53080           CALL    FRMEVL>         ;PICK UP FIRST NON-RELATIONAL
3646                            53100                                   ;CHARACTER AGAIN AND INTERPRET FORMULA
3647                            53120                                   ;ANSWER LEFT IN FAC
3648  004333' 001000  000576    53140   IFE     LENGTH=2,<CPI   44      ;A COMMA?
3649  004334' 000000  000054
3650  004335' 001000  002514    53160           CZ      CHRGTR>         ;IF SO SKIP IT
3651  004336' 000000  003426'
3652  004337' 000000  004326'
3653                            53180   IFN     LENGTH,<
3654  004338' 001000  000376    53200           CPI     GOTOTK          ;ALLOW "GOTO" AS WELL
3655  004339' 000000  000210
3656  004340' 001000  002516    53220           JZ      OKGOTO>
3657  004341' 000000  004346'
3658  004342' 000000  004334'
3659  004343' 001000  002517    53240           SYNCHK  THENTK          ;MUST HAVE A THEN
3660  004344' 000000  000345
3661  004345' 001000  002553    53260           DCX     H
3662  004346'                   53280   OKGOTO:
3663                            53300   IFE     LENGTH,<
3664                            53320           POP     PSW
3665                            53340           POPR                    ;POP OFF NUMBER
3666                            53360   IFN     STRING,<
3667                            53380           XTHL>                   ;COMPARE FORMULA TYPES
```

```
                                53400   IFE     STRING,<PUSH    H>      ;SAVE THE TEXT POINTER
                                53420           PUSH    PSW             ;RESAVE RELATIONAL MEMORIES
                                53440   IFN     STRING,<
                                53460           LDA     VALTYP          ;GET VALUE TYPE
                                53480           CMP     M               ;(M) HAS OLD VALTYP ARE THEY =?
                                53500           JNZ     TMERR           ;IF NOT IIS A TYPE ERROR
                                53520           ORA     A               ;SEE WHAT TYPE IT WAS
                                53540           JZ      NUMCMP          ;ZERO MEANS IT WAS NUMERIC
                                53560           CALL    STRCMP          ;MUST BE STRING,SO STRING COMPARE
                                53580           JMP     SKPNCM>         ;SKIP OVER NUMERIC COMPARE
                                53600   NUMCMP: CALL    FCOMP           ;COMPARE THE 2 SIDE OF THE RELATION STATEMENT
                                53620   SKPNCM: INR     A               ;BUILD RELATIONAL BITS
                                53640           RAL                     ;LESS=4 EQUAL=2 GREATER=1
                                53660                                   ;SINCE CARRY IS ON ONLY IN
                                53680                                   ;377  CASE (FCOMP & STRCMP)
                                53700           POP     B               ;POP OFF WHAT RELATIONAL OPERATOR WAS
                                53720           ANA     B>              ;SEE IF WE MATCHED
                                53740   IFN     LENGTH,<
                                53742   IFE     LENGTH=2,<
                                53744           CALL    VSIGN##>
                                53746   IFN     LENGTH=2,<
                                53760           FSIGN>>                 ;0=FALSE ALL OTHERS=TRUE
                                53780   IFE     LENGTH,<
                                53800           POP     H>              ;POP OFF TEXT POINTER
                                53820   IFE     LENGTH=2,<
                                53840           JZ      FALSIF>         ;HANDLE POSSIBLE "ELSE"
                                53860   IFN     LENGTH=2,<
                                53880           JZ      REM>            ;IF TEST FAILED => JUST SKIP REST OF THE LINE
                                53900   DOCOND: CHRGET                  ;PICK UP THE FIRST LINE # CHARACTER
                                53920           JC      GOTO            ;LINE NUMBER MEANS "GOTO"
                                53940           JMP     GONE3           ;INTERPRET NEW STATEMENT
                                53960   IFE     LENGTH=2,<
                                53980   >
                                54000   ; "ELSE" HANDLER, HERE ON FALSE "IF" CONDITION
                                54020   ;
                                54040   FALSIF: MVI     D,1             ;NUMBER OF "ELSE"S THAT MUST
                                54060                                   ;BE SEEN, "DATA" INCREMENTS THIS
                                54080                                   ;COUNT EVERY TIME AN "IF" IS SEEN
                                54100   SKPMRF: CALL    DATA            ;SKIP A STATEMENT
                                54120                                   ;"1" IS STUCK IN FRONT OF "ELSE"S
                                54140                                   ;SO THAT "DATA" WILL STOP BEFORE "ELSE" CLAUSES
```



```
                                54160           ORA     A               ;END OF LINE?
                                54180           RZ                      ;IF SO, NO "ELSE" CLAUSE
                                54200           CHRGET                  ;SEE IF WE HIT AN "ELSE"
                                54220           CPI     ELSETK
                                54240           JNZ     SKPMRF          ;NO, STILL IN THE "THEN" CLAUSE
                                54260           DCR     D               ;DECREMENT THE NUMBER OF "ELSE"S THAT
                                54280                                   ;MUST BE SEEN
                                54300           JNZ     SKPMRF          ;SKIP MORE IF HAVEN'T SEEN
                                54320                                   ;ENOUGH
                                54340           JMP     DOCOND>         ;FOUND THE RIGHT "ELSE" => GO EXECUTE
                                54360   PAGE
```

```
                                        54400   SUBTTL PRINT CODE
                                        54420
                                        54440   IFN     LPTSW,<
                                        54460   LPRINT: MVI     A,1             ;SAY NON ZERO
                                        54480           STA     PRTFLG>         ;SAVE AWAY
004407' 001000  000053                  54500   NEWCHR: DCX     H
004410' 001000  000327                  54520   MORPR:  CHRGET                  ;GET ANOTHER CHARACTER
004411' 001000  000312                  54540   PRINT:  JZ      CRDO            ;IF WE SEE A TERMINATOR
004412' 000000  004525'
004413' 004000  004485'
                                        54560                                   ;GO TYPE A CRLF
004414' 001000  000310                  54580   PRINTC: RZ                      ;HERE AFTER SEEING TAB(X) OR , OR ;
                                        54600                                   ;IN WHICH CASE A TERMINATOR DOES NOT
                                        54620                                   ;MEAN WE SHOULD TYPE A CRLF
                                        54640                                   ;BUT JUST RETURN
                                        54660   IFE     STRING,<
                                        54680           CPI     34              ;A TERMINATING QUOTE?
                                        54700           CZ      STRJ.1
                                        54720           JZ      NEWCHR>         ;A QUOTATION -- JUST PRINT IT
004415' 001000  000376                  54740           CPI     TABTK
004416' 000000  000240
004417' 001000  000312                  54760           JZ      TABER           ;THE TAB FUNCTION?
004420' 000000  004577'
004421' 000000  004412'
                                        54780   IFN     LENGTH,<
004422' 001000  000376                  54800           CPI     SPCTK
004423' 000000  000242
004424' 001000  000312                  54820           JZ      TABER>          ;THE SPC FUNCTION?
004425' 000000  004577'
004426' 000000  004420'
004427' 001000  000345                  54840           PUSH    H               ;SAVE THE TEXT POINTER
004430' 001000  000376                  54860           CPI     44
004431' 000000  000054
004432' 001000  000312                  54880           JZ      COMPRT          ;IS IT A COMMA?
004433' 000000  004553'
004434' 000000  004425'
004435' 001000  000376                  54900           CPI     59              ;IS IT A ";"
004436' 000000  000073
004437' 001000  000320                  54920           JZ      NOTABH
004440' 000000  004637'
004441' 000000  004433'
004442' 001000  000301                  54940           POP     B               ;GET RID OF OLD TEXT POINTER
004443' 001000  000315                  54960           CALL    FRMEVL          ;BACK UP ONE CHARACTER AND READ THAT ONE
004444' 000000  005336'
004445' 000000  004440'
                                        54980                                   ;AGAIN SO THE CONDITION CODES ARE RIGHT
                                        55000                                   ;EVALUATE THE FORMULA
004446' 001000  000053                  55020           DCX     H               ;BACKUP FROM TERMINATOR
004447' 001000  000345                  55040           PUSH    H               ;SAVE TEXT POINTER
                                        55060   IFN     STRING,<
                                        55080   IFE     LENGTH=2,<
004450' 001000  000315                  55084           CALL    GETYPE          ;SEE IF WE HAVE A STRING
004451' 000000  006307'
```



```
004452' 004000  004444'
004453' 001000  000312                  55086           JZ      STRJON>         ;IF SO, PRINT SPECIALY
004454' 004000  004587'
004455' 000000  004451'
                                        55068   IFN     LENGTH=2,<
                                        55080           LDA     VALTYP
                                        55100           ORA     A
                                        55120           JNZ     STRJON>>        ;WAS IT A STRING FORMULA?
004456' 001000  000315                  55140           CALL    FOUT            ;MAKE A NUMBER INTO A STRING
004457' 000000  000000*
004460' 000000  004454'
                                        55160   IFE     STRING,<CALL    STROUT> ;IF STRINGS OFF JUST OUTPUT IT
                                        55180   IFN     STRING,<
004461' 001000  000315                  55200           CALL    STRLIT          ;MAKE IT  A STRING
004462' 002000  007637'
004463' 000000  004457'
004464' 001000  000052                  55220           LHLD    FACLO           ;GET THE POINTER
004465' 002000  001637'
004466' 002000  0046BE'
                                        55240   IFN     LPTSW,<
                                        55260           LDA     PRTFLG
                                        55280           ORA     A
004470' 001000  000312                  55300           JZ      ISTTY           ;LPT OR TTY?
                                        55320           LDA     LPTPOS
                                        55340           ADD     M
004471' 002000  004C26                  55360           CPI     LPTLEN          ;WILL THIS NUMBER OVERLAP?
004472' 001000  000376                  55380           JMP     LINCHK
004473' 002000  002014
004474' 004474'                         55400   ISTTY:>
004475' 001000  000072                  55420           LDA     TTYPOS          ;SEE WHERE WE ARE
004476' 000000  000047'
004477' 000000  004465'
004500' 001000  000206                  55440           ADD     M               ;ADD THIS LENGTH
004501' 001000  000376                  55460           CPI     LINLEN          ;SEE IF GREATER THAN THE LINE LENGTH
004502' 000000  000240
004503' 001000  000076                  55480   LINPT3:=:.-1
004504' 000000  000040                  55500   LINCHK: CNC     CRDO            ;IF SO CRLF
004505' 001000  000337                  55520           CALL    STRPRT>         ;PRINT THE NUMBER
                                        55540           MVI     A," "           ;ALWAYS END WITH A SPACE
                                        55560           OUTCHR
                                        55580   IFN     STRING,<
                                        55590   IFE     LENGTH=2,<
004506' 001000  000267                  55595           ORA     A               ;TURN OFF THE ZERO FLAG
004507' 001000  000314                  55597   STRJON: CZ      STRPRT>
004510' 000000  007746'
004511' 002000  004501'
                                        55598   IFN     LENGTH=2,<
                                        55599           XRA     A
                                        55600   STRJON: CNZ     STRPRT>>        ;JUST PRINT STRINGS
```

```
                                                                
 3846  004512' 001000  000341        55620           POP     H
 3847  004513' 001000  000303        55640           JMP     MORPR           ;PRINT SOME MORE
 3848  004514' 000000  0044,0'
 3849  004515' 000000  0045,0'
 3850  004516' 001000  000066        55660   FININL: MVI     M,0             ;PUT A ZERO AT THE END OF BUF
 3851  004517' 000020  000000
 3852  004520' 001000  000041        55680           LXI     H,BUFMIN        ;SETUP POINTER
 3853  004521' 000000  001430'
 3854  004522' 000000  0045,4'
 3855                                 55700   CRDO1:  IFN     LPTSW,<
 3856                                 55720           LDA     PRTFLG
 3857                                 55740           ORA     A
 3858                                 55760           JNZ     PRINTW>
 3859  004524' 001000  000076        55780           MVI     A,13
 3860  004525' 000000  000015
 3861  004526' 001000  000062        55800           STA     TTYPOS          ;MAKE TTYPOS LESS THAN LINE LENGTH
 3862  004527' 000000  000047'
 3863  004528' 000000  004521'
 3864                                 55820                                   ;SO CRDO DOESN'T GET CALLED AGAIN
 3865  004530' 001000  000337        55840           OUTCHR
 3866  004531' 001000  000076        55860           MVI     A,10
 3867  004532' 000000  0000,2
 3868  004533' 001000  000337        55880           OUTCHR
 3869  004534' 001000  000072        55900   CRFIN:  LDA     NULCNT          ;GET NUMBER OF NULLS
 3870  004535' 000000  000046'
 3871  004536' 000000  004526'
 3872  004537' 001000  000075        55920   PRTNUL: DCR     A
 3873  004540' 001000  000062        55940           STA     TTYPOS          ;EVENTUALLY SETUP TTYPOS=0
 3874  004541' 000000  000047'
 3875  004542' 000000  004535'
 3876  004543' 001000  000310        55960           RZ                      ;ALL NULLS DONE (A)=0
 3877                                 55980                                   ;SOME ROUTINES DEPEND ON CRLO
 3878                                 56000                                   ;AND CRFIN8 RETURN (A)=0 AND Z  TRUE
 3879  004544' 001000  000365        56020           PUSH    PSW             ;SAVE THE COUNT
 3880  004545' 001000  000257        56040           XRA     A               ;(A)= A NULL
 3881  004546' 001000  000337        56060           OUTCHR                  ;SEND IT OUT
 3882  004547' 001000  000361        56080           POP     PSW             ;RESTORE THE COUNT
 3883  004550' 001000  000303        56100           JMP     PRTNUL          ;LOOP PRINTING NULLS
 3884  004551' 000000  004537'
 3885  004552' 000000  004541'
 3886                                 56120   IFE     STRING,<
 3887                                 56140   STROU1: INX     H
 3888                                 56160   STROUT: MOV     A,M
 3889                                 56180           ORA     A               ;CHECK FOR END OF LINE
 3890                                 56200           RZ                      ;LET IT END THAT WAY
 3891                                 56220           INX     H
 3892                                 56240           CPI     34              ;A TERMINATING QUOTE?
 3893                                 56260           RZ                      ;DONE IF SO
 3894                                 56280           OUTCHR                  ;PRINT THE CHARACTER IN (A)
 3895                                 56300           CPI     CR              ;IF IT'S A CARRIAGE RETURN
 3896                                 56320           CZ      CRDO            ;TYPE LINE-FEED AND SET (TTYPOS)=0
 3897                                 56340           JMP     STROUT>         ;PRINT MORE CHARACTERS
 3898
```



```
                                 56360   COMPRT: IFN     LPTSW,<
                                 56400           LDA     PRTFLG          ;OUTPUT TO THE LINE PRINTER?
                                 56420           ORA     A               ;NON-ZERO MEANS YES
                                 56440           JZ      ISCTTY          ;NO, DO TELETYPE COMMA
                                 56460           LDA     LPTPOS          ;GET LINE PRINTER POSITION
                                 56480   NLPPOS==<<<LPTLEN/CLMWID>-1>*CLMWID>;POSITION BEYOND WHICH THERE ARE
                                 56500                                   ;NO MORE COMMA FIELDS, SO
                                 56520           CPI     NLPPOS          ;COMMA JUST DOES A "CRDO"
                                 56540           JMP     CHKCOM          ;USE TELETYPE CHECK
                                 56560   ISCTTY:>
 3909  004553' 001000  000072        56580           LDA     TTYPOS          ;GET TELETYPE POSITION
 3910  004554' 000000  000047'
 3911  004555' 000000  004551'
 3912                  000070        56600   NOTPOS==<<<LINLEN/CLMWID>-1>*CLMWID>;POSITION BEYOND WHICH THERE ARE
 3913                                 56620                                   ;NO MORE COMMA FIELDS
 3914  004556' 001000  000376        56640           CPI     NOTPOS          ;SO ALL COMMA DOES IS A "CRDO"
 3915  004557' 000000  002070
 3916                  004557'        56660   LINPT4==I,=1                    ;FIXED UP BY "TERMINAL WIDTH" QUESTION
 3917  004560' 001000  000324        56680   CHKCOM: CNC     CRDO            ;TYPE CR,F
 3918  004561' 000000  004523'
 3919  004562' 000000  004554'
 3920  004563' 001000  000322        56700           JNC     NOTABR          ;AND QUIT IF BEYOND THE LAST COMMA FIELD
 3921  004564' 000000  004637'
 3922  004565' 000000  004561'
 3923  004566' 001000  000326        56720   MORCOM: SUI     CLMWID          ;GET (A) MODULUS CLMWID
 3924  004567' 000000  000006
 3925  004570' 001000  000322        56740           JNC     MORCOM
 3926  004571' 000000  004566'
 3927  004572' 000000  004564'
 3928  004573' 001000  000257        56760           CMA                     ;WE WANT TO  FILL
 3929                                 56780                                   ;THE PRINT POSITION OUT
 3930                                 56800                                   ;TO AN EVEN CLMWID, SO
 3931                                 56820                                   ;WE PRINT CLMWID-(A) MOD CLMWID SPACES
 3932  004574' 001000  000303        56840           JMP     ASPA2           ;GO PRINT (A)+1 SPACES
 3933  004575' 000000  004626'
 3934  004576' 000000  004571'
 3935
 3936  004577'                        56860   TABER1:
 3937  004577' 001000  000365        56900   IFN     LENGTH,<PUSH    PSW>    ;REMEMBER IF (A)=SPCTK OR TABTK
 3938                                 56920   IFE     LENGTH,<
 3939                                 56940           CALL    INTIDX>         ;INTEGERIZE A FORMULA INTO (D,E)
 3940                                 56960   IFN     LENGTH,<
 3941  004600' 001000  000315        56980           CALL    GTBYTC>         ;GET VALUE INTO (E)
 3942  004601' 000000  0110,7'
 3943  004602' 000000  004575'
 3944  004603' 001000  0003,7        57000           SYNCHR  ")"
 3945  004604' 000000  000051
 3946  004605' 001000  000053        57020           DCX     H
 3947                                 57040   IFN     LENGTH,<
 3948  004606' 001000  000361        57060           POP     PSW             ;GET BACK SPCTK OR TABTK
 3949  004607' 001000  000376        57080           CPI     SPCTK>          ;WAS IT SPCTK?
 3950  004610' 000000  000042
 3951  004611' 001000  000345        57100           PUSH    H               ;SAVE THE TEXT POINTER
```

```
                                                57120   IFN     LENGTH,<
                                                57140           MOV     A,E             ;FOR "SPC" PUT THE FORMULA
                                                57100           JZ      ASPAC>          ;VALUE IN [A]
                                                57180   IFN     LPTSW,<
                                                57200           LDA     PRTFLG          ;LINE PRINTER OR TTY?
                                                57220           ORA     A               ;NON-ZERO MEANS LPT
                                                57240           JZ      TTYIS1
                                                57260           LDA     LPTPOS          ;GET LINE PRINTER POSITION
                                                57280           JMP     DOSIZT>
0046167' 001000  000072         57300   TTYIS1: LDA     TTYPOS          ;GET TELETYPE PRINT POSITION
0046177' 000000  000047'
0046207' 000000  004614'
0046217  001000  000037         57320   DOSIZT: CMA                     ;PRINT [E]=[A] SPACES
0046227' 001000  000203         57340           ADD     E
0046237' 001200  000322         57360           JNC     NOTABR          ;IF NEGATIVE, DON'T PRINT ANY
0046297' 000000  004637'
0046257' 000000  004617'
                                                57380                   ;SPACES
0046267' 001000  000074         57400   ASPAC:  INR     A
0046277' 001000  000107         57420   ASPAC1: MOV     B,A             ;[B]=NUMBER OF SPACES TO PRINT
0046307' 001000  000076         57440           MVI     A,' '           ;[A]=SPACE
0046317' 000000  000040
0046327' 001000  000337         57460   REPOUT: OUTCHR                  ;PRINT [A]
0046337' 001000  000005         57480           DCR     B               ;DECREMENT THE COUNT
0046347' 001000  000322         57500           JNZ     REPOUT
0046357' 000000  00465E'
0046367' 000001  004624'
0046377' 001000  000341         57520   NOTABR: POP     H               ;PICK UP TEXT POINTER
0046407' 001000  000327         57540           CHRGET                  ;AND THE NEXT CHARACTER
0046417' 001000  000303         57560           JMP     PRINTC          ;AND SINCE WE JUST PRINTED
0046427' 000000  004414'
0046437' 000000  004635'
                                                57580                   ;SPACES, DON'T CALL CRDO
                                                57600                   ;IF IT'S THE END OF THE LINE
                                                57620   PAGE
```



```
                                                57640   SUBTTL INPUT AND READ CODE
                                                57660   IFN     LENGTH,<
0046447' 000000  000077         57680   TRYAGN: DC"?REDO FROM START"
0046457' 000000  000122
0046467' 000000  000105
0046477' 000000  000104
0046507' 000000  000117
0046517' 000000  000122
0046527' 000000  000040
0046537' 000000  000123
0046547' 000000  000124
0046557' 000000  000101
0046567' 000000  000122
0046577' 000000  000124
0046607' 000000  000040
0046617' 000000  000015         57700           ACRLF
0046627' 000000  000012
0046637' 000000  000000         57720           0
                                                57740   ;
                                                57760   ; HERE WHEN THE DATA THAT WAS TYPED IN OR IN "DATA" STATEMENTS
                                                57780   ; IS IMPROPERLY FORMATTED.  FOR "INPUT" WE START AGAIN,
                                                57800   ; FOR "DATA" WE GIVE A SYNTAX ERROR AT THE DATA LINE
                                                57820   ;
0046647' 001000  000072         57840   TRMNOK: LDA     FLGINP          ;WAS IT READ OR INPUT?
0046657' 000000  001602'
0046667' 000000  004644'
                                                57860           ORA     A               ;ZERO=INPUT
0046677' 001000  000267         57880           JNZ     DATSNE          ;GIVE ERROR AT DATA LINE
0046707' 000000  002644'
0046717' 000000  004670'
                                                57900           POP     B               ;GET RID OF THE POINTER INTO THE VARIABLE LIST
0046727' 001000  000301
0046737' 001000  000041         57920           LXI     H,TRYAGN
0046747' 000000  004644'
0046757' 000000  004674'
0046767' 001000  000315         57940           CALL    STROUT          ;PRINT "?REDO FROM START"
0046777' 000000  007745'
0047007' 000000  004700'
0047017' 001000  000052         57960           LHLD    TEMP            ;START ALL OVER BY GOING BACK
0047027' 000000  001603'
0047037' 000000  004703'
0047047' 001000  000311         57980           RET>                    ;TO NEWSTT POINTING AT THE START OF
                                                57982                   ;OF THE "INPUT" STATEMENT
0047117'                        58000   INPUT:
                                58020   IFN     LENGTH,<
0047117' 001000  000376         58040           CPI     3A              ;IS IT A QUOTE?
0047127' 000000  000042
                                58060   IFN     CONTRW,<
0047137' 001000  000076         58080           MVI     A,0             ;BE TALKATIVE
```

```
 4042  004714' 000000  000000
 4043  004715' 001000  000062          58100           STA     CNTWFL>         ;FORCE OUTPUT
 4044  004716' 000000  001341'
 4045  004717' 000000  004706'
 4046  004720' 001000  000302          58120           JNZ     NOTQTI          ;IF NOT NO MESSAGE
 4047  004721' 000000  004735'
 4048  004722' 000000  004716'
 4049                                  58140   IFN     STRING,<
 4050  004723' 001000  000315          58160           CALL    STRLTI          ;MAKE THE MESSAGE A STRING
 4051  004724' 000000  007640'
 4052  004725' 000000  004721'
 4053  004726' 001000  000317          58180           SYNCHK  59              ;MUST END WITH SEMI-COLON
 4054  004727' 000000  000073
 4055  004730' 001000  000345          58200           PUSH    H               ;REMEMBER WHERE IT ENDED
 4056  004731' 001000  000315          58220           CALL    STRPRT          ;PRINT IT OUT
 4057  004732' 000000  007746'
 4058  004733' 000000  004724'
 4059  004734' 001000  000341          58240           POP     H>
 4060                                  58260   IFE     STRING,<
 4061                                  58280           CALL    STRUUI
 4062                                  58300           SYNCHK  59>>            ;ENDS WITH SEMI-COLON
 4063  004735' 001000  000345          58320   NOTQTI: PUSH    H
 4064                                  58340   IFE     FUNCTS,<
 4065                                  58360           LHLD    CURLIN          ;IS IT DIRECT?
 4066                                  58380           MVI     E,ERRID         ;IF SO "ILLEGAL DIRECT" ERROR
 4067                                  58400           INX     H               ;DIRECT MEANS THAT
 4068                                  58420           MOV     A,L             ;CURLIN=65535
 4069                                  58440           ORA     H               ;ADDED ONE AND GOT 0?
 4070                                  58460           JZ      ERROR>          ;IF SO,GO COMPLAIN
 4071                                  58480   IFN     FUNCTS,<
 4072  004736' 001000  000315          58500           CALL    ERRDIR>         ;USE COMMON ROUTINE SINCE DEF
 4073  004737' 000000  007532'
 4074  004740' 000000  004732'
 4075                                  58520                                   ;DIRECT IS ALSO ILLEGAL
 4076  004741' 001000  000315          58540   GETAGN: CALL    QINLIN          ;TYPE "?" AND INPUT A LINE OF TEXT
 4077  004742' 000000  002522'
 4078  004743' 000000  004737'
 4079  004744' 001000  000043          58560   IFN     LENGTH,<INX     H       ;IF NO INPUT WE QUIT
 4080  004745' 001000  000176          58580           MOV     A,M
 4081  004746' 001000  000267          58600           ORA     A
 4082  004747' 001000  000053          58620           DCX     H
 4083  004750' 001000  000301          58640           POP     B               ;TAKE OFF SINCE MAYBE LEAVING
 4084  004751' 001000  000312          58660           JZ      STPEND          ;IF EMPTY LEAVE
 4085  004752' 000000  003540'
 4086  004753' 000000  004742'
 4087  004754' 001000  000305          58680           PUSH    B>              ;PUT BACK  SINCE DIDN'T LEAVE
 4088  004755' 001000  000303          58700           JMP     INPCON
 4089  004756' 000000  004765'
 4090  004757' 000000  004752'
 4091  004760' 001000  000345          58720   READ:   PUSH    H               ;SAVE THE TEXT POINTER
 4092  004761' 001000  000052          58740           LHLD    DATPTR          ;GET LAST DATA LOCATION
 4093  004762' 000000  001627'
 4094  004763' 000000  004756'
```



```
 4095  004764' 001000  000366          58760           XWD     ^O1000,^O366    ;"ORI" TO SET [A] NON-ZERO
 4096  004765' 001000  000257          58780   INPCON: XRA     A               ;SET FLAG THAT THIS IS AN INPUT
 4097  004766' 001000  000062          58800           STA     FLGINP          ;STORE THE FLAG
 4098  004767' 000000  001602'
 4099  004770' 000000  004762'
 4100                                  58820   ;
 4101                                  58840   ; IN THE PROCESSING OF DATA AND READ STATEMENTS:
 4102                                  58860   ; ONE POINTER POINTS TO THE DATA (IE THE NUMBERS BEING FETCHED)
 4103                                  58880   ; AND ANOTHER POINTS TO THE LIST OF VARIABLES
 4104                                  58900   ;
 4105                                  58920   ; THE POINTER INTO THE DATA ALWAYS STARTS POINTING TO A
 4106                                  58940   ; TERMINATOR -- A , S OR END-OF-LINE
 4107                                  58960   ;
 4108  004771' 001000  000343          58980           XTHL                    ;(H,L)=VARIABLE LIST POINTER
 4109                                  58982                                   ;DATA POINTER GOES ON THE STACK
 4110  004772' 001000  000001          59000           XWD     ^O1000,1        ;"LXI B," OVER THIS CHECK
 4111  004773' 001000  000317          59020   LOPJT2: SYNCHK  44              ;MAKE SURE THERE IS A ","
 4112  004774' 000000  000054
 4113  004775' 001000  000315          59040           CALL    PTRGET          ;READ THE VARIABLE LIST
 4114  004776' 000000  006585'
 4115  004777' 000000  004767'
 4116                                  59060                                   ;AND GET THE POINTER TO A VARIABLE INTO (D,E)
 4117  005000' 001000  000343          59080           XTHL                    ;PUT THE VARIABLE LIST POINTER ONTO THE
 4118                                  59100                                   ;STACK AND TAKE THE
 4119                                  59120                                   ;DATA LIST POINTER OFF
 4120                                  59140   ;
 4121                                  59160   ; NOTE AT THIS POINT WE HAVE A VARIABLE WHICH WANTS DATA
 4122                                  59180   ; AND SO WE MUST GET DATA OR COMPLAIN
 4123                                  59200   ;
 4124  005001' 001000  000325          59220           PUSH    D               ;SAVE THE POINTER TO THE VARIABLE WE
 4125                                  59240                                   ;ARE ABOUT TO SET UP WITH A VALUE
 4126  005002' 001000  000176          59260           MOV     A,M             ;SINCE THE DATA LIST POINTER ALWAYS POINTS
 4127                                  59280                                   ;AT A TERMINATOR LETS READ THE
 4128                                  59300                                   ;TERMINATOR INTO [A] AND SEE WHAT
 4129                                  59320                                   ;IT IS
 4130  005003' 001000  000376          59340           CPI     44
 4131  005004' 000000  000054
 4132  005005' 001000  000312          59360           JZ      DATBK           ;A COMMA SO A VALUE MUST FOLLOW
 4133  005006' 000000  005025'
 4134  005007' 000000  004776'
 4135                                  59380   IFE     LENGTH,<
 4136                                  59400           ORA     A               ;IN THE 4K VERSION
 4137                                  59420           JNZ     SNERR>          ;DATA MUST BE ALONE ON A LINE
 4138  005010' 001000  000072          59440           LDA     FLGINP          ;SEE WHAT TYPE OF STATEMENT THIS WAS
 4139  005011' 000000  001602'
 4140  005012' 000000  005026'
 4141  005013' 001000  000267          59460           ORA     A
 4142                                  59480   IFE     LENGTH,<
 4143                                  59500           INX     H               ;POINT AT POINTER TO NEXT LINE
 4144                                  59520           JNZ     DATEND>         ;IF IT IS A READ GO
 4145                                  59540                                   ;SEARCH FOR ANOTHER DATA STATEMENT
 4146  005014' 001000  000302          59560   IFN     LENGTH,<JNZ     DATLOP>
 4147  005015' 000000  005163'
```

```
4148  005016' 000000  005011'
4149  005017' 001000  000076          59580           MVI     A,"?"
4150  005020' 000000  000077
4151  005021' 001000  000337          59600           OUTCHR                  ;DOUBLE PROMPT WHEN WE NEED MORE INPUT
4152  005022' 001000  000315          59620           CALL    GINLIN          ;GET A WHOLE LINE AFTER TYPING "?"
4153  005023' 000000  002522'
4154  005024' 000000  005015'
4155                                   59640                                   ;THE DATA NOW STARTS AT THE BEGINNING
4156                                   59660                                   ;OF THE BUFFER
4157                                   59680                                   ;AND GINLIN LEAVES [H,L]=BUF
4158  005025'                          59700   DATBK1:
4159                                   59720   IFE     LENGTH,<POP     D       ;POP OFF POINTER TO THE VARIABLE
4160                                   59740           INX     H               ;FUDGE CHARACTER POINTER
4161                                   59760           CALL    REDINP>         ;READ A VALUE USING "LET" CODE
4162                                   59780   IFN     LENGTH,<
4163                                   59800   IFN     STRING,<
4164  005025' 001000  000076          59820           LDA     VALTYP          ;SEE IF ITS NUMERIC OR STRING
4165  005026' 000000  001543'
4166  005027' 000000  005023'
4167                                   59840   IFE     LENGTH-2,<
4168  005030' 001000  000376          59860           CPI     3               ;IS IT A STRING ?
4169  005031' 000000  000003
4170  005032' 001000  000365          59880           PUSH    PSW             ;SAVE THE TYPE INFORMATION
4171  005033' 001000  000302          59900           JNZ     NUMINS>         ;IF NUMERIC,USE FIN TO GET IT
4172  005034' 000000  005070'
4173  005035' 000000  005026'
4174                                   59920   IFN     LENGTH-2,<
4175                                   59940           ORA     A
4176  005036' 001000  000310          59960           RZ                      ;INPUT A NUMBER IF NUMERIC
4177                                   59980                                   ;ONLY THE VARIABLE TYPE IS
4178                                   60000                                   ;CHECKED SO AN UNQUOTED STRING
4179                                   60020                                   ;CAN BE ALL DIGITS
4180  005037' 001000  000027          60040           CHRGET
4181  005040' 001000  000127          60060           MOV     D,A             ;ASSUME QUOTED STRING
4182  005041' 001000  000107          60080           MOV     B,A             ;SETUP TERMINATORS
4183  005042' 001000  000376          60100           CPI     34              ;QUOTE ?
4184  005043' 000000  000042
4185  005044' 001000  000312          60120           JZ      NOWGET          ;TERMINATORS OK
4186  005045' 000000  005053'
4187  005046' 000000  005034'
4188  005047' 001000  000026          60140           MVI     D,":"           ;UNQUOTED STRING TERMINATORS
4189  005050' 000000  000072
4190  005051' 001000  000026          60160           MVI     B,44            ;ARE COLON AND COMMA
4191  005052' 000000  000054
4192                                   60180           DCX     H               ;BACKUP SINCE START CHARACTER MUST BE INCLUDED
4193  005053' 001000  000053
4194                                   60200                                   ;IN THE QUOTED STRING CASE WE DON'T WANT TO
4195                                   60220                                   ;INCLUDE THE STARTING OR ENDING QUOTE
4196  005054' 001000  000315          60240   NOWGET: CALL    STRLT2          ;MAKE A STRING DESCRIPTOR FOR THE VALUE
4197  005055' 000000  007643'
4198  005056' 000000  005044'
4199                                   60242                                   ;AND COPY IF NECESSARY
4200                                   60260   IFE     LENGTH-2,<
```



```
4201  005057' 001000  000361          60280   DOASIG: POP     PSW>            ;POP OFF THE TYPE INFORMATION
4202  005060' 001000  000353          60320           XCHG                    ;[D,E]=TEXT POINTER
4203  005061' 001000  000041          60340           LXI     H,STRUN2        ;RETURN LOC
4204  005062' 000000  005077'
4205  005063' 000000  005054'
4206  005064' 001000  000343          60360           XTHL                    ;[H,L]=PLACE TO STORE VARIABLE VALUE
4207  005065' 001000  000325          60380           PUSH    D               ;TEXT POINTER GOES ON
4208  005066' 001000  000303          60400           JMP     INPCOM>         ;DO ASSIGNMENT
4209  005067' 000000  004137'
4210  005070' 000000  005061'
4211  005071' 001000  000327          60420   NUMINS: CHRGET
4212  005072' 001000  000315          60440           CALL    FIN
4213  005073' 000000  000000*
4214  005075' 000000  005066'
4215                                   60460   IFE     LENGTH-2,<
4216  005076' 001000  000303          60480           JMP     DOASIG>         ;ASSIGNMENT IS COMPLICATED
4217  005077' 000000  005056'
4218  005078' 000000  005072'
4219                                   60500                                   ;EVEN FOR NUMERICS SO USE THE "LET" CODE
4220                                   60520   IFN     LENGTH-2,<
4221                                   60540           XTHL                    ;[H,L] GET POINTER AT VARIABLE
4222                                   60560           CALL    MOVMF           ;DO THE ASSIGNMENT
4223                                   60580           POP     H>>             ;GET BACK THE TEXT POINTER
4224  005077'                          60600   STRDN2:
4225                                   60620   IFN     LENGTH,<
4226  005077' 001000  000053          60640           DCX     H
4227  005100' 001000  000327          60660           CHRGET
4228  005101' 001000  000312          60680           JZ      TRMOK
4229  005102' 000000  005111'
4230  005103' 000000  005075'
4231  005104' 001000  000376          60700           CPI     44
4232  005105' 000000  000054
4233  005106' 001000  000302          60720           JNZ     TRMNOK>         ;ENDED PROPERLY?
4234  005107' 000000  004667'
4235  005110' 000000  005102'
4236  005111' 001000  000343          60740   TRMOK:  XTHL
4237  005112' 001000  000053          60760           DCX     H               ;LOOK AT TERMINATOR
4238  005113' 001000  000327          60780           CHRGET                  ;AND SET UP CONDITION CODES
4239  005114' 001000  000302          60800           JNZ     LOPDT2          ;NOT ENDING, CHECK FOR COMMA
4240  005115' 000000  004773'
4241  005116' 000000  005107'
4242                                   60820                                   ;AND GET ANOTHER VARIABLE
4243                                   60840                                   ;TO FILL WITH DATA
4244
4245  005117' 001000  000321          60860           POP     D               ;POP OFF THE POINTER INTO DATA
4246  005120' 001000  000072          60900           LDA     FLGINP          ;FETCH THE STATEMENT TYPE FLAG
4247  005121' 000000  001662'
4248  005122' 000000  005115'
4249  005123' 001000  000267          60920           ORA     A
4250                                   60940   IFE     LENGTH,<RZ>             ;DON'T UPDATE DATPTR IF IT WAS AN
4251                                   60960                                   ;INPUT STATEMENT
4252  005124' 001000  000353          60980           XCHG
4253  005125' 001000  000302          61000           JNZ     RESFIN          ;UPDATE DATPTR
```

```
                                                61920   IFN     LENGTH,<
0052D4' 001000  000045                          61600           INX     H>
0052D5' 021000  000367                          61620           PUSHM                   ;SAVE POINTER TO THE NEXT STATEMENT
0052D7' 011000  0B0171                          61640           MOV     A,C             ;SEE IF WE ARE AT THE END
0052D8' 001000  000060                          61660           ORA     A
0052DA' 001000  000036                          61680           MVI     E,ERROD         ;NO DATA IS ERROR ERROD
0052DC' 000000  000004
0052DD' 001000  000031                          61700           JZ      ERROR           ;IF SO COMPLAIN
0052E0' 040000  0B210E'
0052E2' 000000  005170'
                                                61720   IFE     LENGTH,<INX     H>      ;SKIP PAST LINE #
0052E3' 001000  000361                          61740   IFN     LENGTH,<POP     B
0052E4' 011000  002136                          61760           MOV     E,M             ;GET DATA LINE #
0052E5' 001000  000043                          61780           INX     H
0052E6' 011000  000126                          61800           MOV     D,M
0052E7' 011000  000353                          61820           XCHG
0052E8' 001000  000042                          61840           SHLD    DATLIN
0052EB' 001000  000000  001577'
0052EE' 000000  005201'
                                                61860           XCHG>                   ;RESTORE TEXT POINTER
0052F1' 011000  000353                          61880   NOWLIN: CMPGET                  ;GET THE STATEMENT TYPE
0052F2' 021000  000527                          61900           CPI     DATATK          ;IS IS "DATA"?
0052F5' 001000  000376
0052F6' 000000  000003
0052F7' 001000  0B0302                          61920           JNZ     DATLOP          ;NOT DATA SO LOOK SOME MORE
0052FA' 040000  0051B5'
0052FC' 000000  005211'
                                                61940   IFE     LENGTH,<POP     B>
0052FF' 001000  000303                          61960           JMP     DATBK           ;CONTINUE READING
005302' 000000  005225'
005305' 000000  005220'
                                                62000   PAGE
```



```
005126' 000000  0E3453'
005127' 000000  0005121'
                                                61020   IFN     LENGTH,<
005130' 021000  000266                          61040           ORA     M               ;COULD HAVE ENDED WITH COMMA OR
                                                61060           ;COLON, BUT SHOULD BE A ZERO
005131' 021000  000041                          61080           LXI     M,EXIGNT        ;TEXT FOR "EXTRA"
005132' 000000  005102'
005133' 000000  0C5126'
005134' 021000  000325                          61100           PUSH    D               ;SAVE THE TEXT POINTER
005135' 021000  000344                          61120           CNZ     STROUT          ;IF WASN'T REAL END SAY SOMETHING
005136' 000000  007743'
005137' 000000  0C5132'
005140' 021000  000341                          61140           POP     M               ;GET BACK THE TEXT POINTER
005141' 021000  000311                          61160           RET
005142' 000000  000077                          61180   EXIGNT: DC"?EXTRA IGNORED"
005143' 000000  000105
005144' 000000  000130
005145' 000000  000124
005146' 000000  000126
005147' 000000  000101
005150' 000000  0C0040
005151' 000000  000111
005152' 000000  000107
005153' 000000  000116
005154' 000000  000117
005155' 000000  000122
005156' 000000  000103
005157' 000000  000104
005160' 000000  000304                          61200           ACRLF
005161' 000000  000015
005162' 000000  000000                          61220           0>
                                                61240   ;
                                                61260   ; SUBROUTINE TO FIND DATA
                                                61280   ; IN THE 8K "DATA" MUST BE AT THE START OF THE LINE
                                                61300   ; SO THE SEARCH IS MADE USING THE LINKS AT THE START OF EACH LINE.
                                                61320   ;
                                                61340   ; IN THE 8K AND EXTENDED THE SEARCH IS MADE BY USING THE EXECUTION CODE
                                                61360   ; FOR DATA TO SKIP OVER STATEMENTS. THE START WORD OF EACH STATEMENT
                                                61380   ; IS COMPARED WITH DATATK, EACH NEW LINE NUMBER
                                                61400   ; IS STORED IN DATLIN SO THAT IF AN ERROR OCCURS WHILE READING
                                                61420   ; DATA THE ERROR MESSAGE WILL GIVE THE LINE NUMBER OF THE
                                                61440   ; ILL-FORMATTED DATA
                                                61460   ;
005163' 041000  000315                          61480   DATLOP: IFN     LENGTH,<CALL    DATA>
005164' 000000  0004072'
005165' 000000  005116'
                                                61500           IFE     LENGTH,<POP     H>
005166' 041000  000267                          61520   DATFND: IFN     LENGTH,<
005167' 021000  000302                          61540           ORA     A
005170' 000000  005214'                         61560           JNZ     NOWLIN>
005171' 000000  005164'
```

```
                                        62360   SUBTTL NEXT CODE
                                        62380   ;
                                        62100   ; NOTE:
                                        62120   ;
                                        62140   ;
                                        62160   ; A FOR ENTRY ON THE STACK HAS THE FOLLOWING FORMAT:
                                        62180   ;
                                        62200   ;
                                        62220   ; LOW ADDRESS
                                        62240   ;       TOKEN (FORTK IN HIGH BYTE)  1 BYTES
                                        62260   ;       A POINTER TO THE LOOP VARIABLE  2 BYTES
                                        62280   ;       A BYTE REFLECTING THE SIGN OF THE INCREMENT 1 BYTE
                                        62300   ;       THE STEP 4 BYTES
                                        62320   ;       THE UPPER VALUE 4 BYTES
                                        62340   ;       THE LINE # OF THE "FOR" STATEMENT 2 BYTES
                                        62360   ;       A TEXT POINTER INTO THE "FOR" STATEMENT 2 BYTES
                                        62380   ; HIGH ADDRESS
                                        62400   ;
                                        62420   ; TOTAL 16 BYTES
                                        62440   ;
4361 005225'                            62460   NEXT:
4362 005225' 001000  000021             62480   IFN     LENGTH,<LXI     D,$CODE>;FOR THE "NEXT"
4363 005226' 000000  000000'
4364 005227' 000000  005223'
4365                                    62500                                   ;STATEMENT WITHOUT ANY ARGS
4366                                    62520                                   ;WE CALL FNDFOR WITH (D,E)=0
4367 005230'                            62540   NEXTC:
4368                                    62560   IFE     LENGTH,<
4369                                    62580           CALL    PTRGET>         ;MUST HAVE A VARIABLE
4370                                    62600   IFN     LENGTH,<
4371 005230' 001000  000304             62620           CNZ     PTRNEI>         ;GET A POINTER TO THE
4372 005231' 000000  006505'
4373 005232' 000000  005226'
4374                                    62640                                   ;LOOP VARIABLE INTO (D,E)
4375 005233' 001000  000042             62660           SHLD    TEMP            ;PUT THE TEXT POINTER
4376 005234' 000000  001604'
4377 005235' 000000  005231'
4378                                    62680                                   ;IN A TEMP LOCATION
4379                                    62700                                   ;IN CASE THE LOOP TERMINATES
4380 005236' 001000  000315             62720           CALL    FNDFOR          ;TRY TO FIND A FOR ENTRY
4381 005237' 000000  001744'
4382 005240' 000000  005234'
4383                                    62740                                   ;ON THE STACK WHOSE VARIABLE NAME
4384                                    62760                                   ;MATCHES THIS ONES
4385                                    62780   IFN     LENGTH,<
4386 005241' 001000  000302             62800           JNZ     NFERR>          ;"NEXT WITHOUT FOR"
4387 005242' 000000  002100'
4388 005245' 000000  005237'
4389 005244' 001000  000371             62820           SPHL                    ;SETUP STACK POINTER BY CHOPPING
4390                                    62840                                   ;AT THIS POINT
4391 005245' 001000  000325             62860           PUSH    D               ;PUT THE VARIABLE PTR BACK ON
4392 005246' 001000  000176             62880           MOV     A,M             ;STEP ONTO THE STACK
4393 005247' 001000  000043             62900           INX     H
```



```
4394 005250' 001000  000365             62920           PUSH    PSW
4395 005251' 001000  000325             62940           PUSH    D               ;PUT THE POINTER TO THE LOOP
4396                                    62960                                   ;VARIABLE ONTO THE STACK
4397                                    62980   IFE     LENGTH,<
4398                                    63000           MVI     E,ERRNF
4399                                    63020           JNZ     ERROR>
4400 005252' 001000  000315             63040           CALL    MOVFM           ;STEP VALUE INTO THE FAC
4401 005253' 000000  000000*
4402 005254' 000000  005242'
4403 005255' 001000  000343             63060           XTHL                    ;PUT THE POINTER INTO THE
4404                                    63080                                   ;FOR ENTRY ONTO THE STACK
4405 005256' 001000  000345             63100           PUSH    H               ;PUT THE POINTER TO THE LOOP
4406                                    63120                                   ;VARIABLE BACK ONTO THE STACK
4407 005257' 001000  000315             63140           CALL    FADDS           ;ADD THE STEP AND LOOP VARIABLE
4408 005260' 000000  000000*
4409 005261' 000000  005253'
4410 005262' 001000  000341             63160           POP     H               ;POP OFF THE POINTER TO
4411                                    63180                                   ;THE LOOP VARIABLE
4412 005263' 001000  000315             63200           CALL    MOVMF           ;MOV FAC INTO LOOP VARIABLE
4413 005264' 000000  000000*
4414 005265' 000000  005260'
4415 005266' 001000  000341             63220           POP     H               ;GET THE ENTRY POINTER
4416 005267' 001000  000315             63240           CALL    MOVRM           ;GET THE FINAL INTO THE REGISTERS
4417 005270' 000000  000000*
4418 005271' 000000  005264'
4419 005272' 001000  000345             63260           PUSH    H               ;SAVE THE ENTRY POINTER
4420 005273' 001000  000315             63280           CALL    FCOMP           ;COMPARE THE NUMBERS
4421 005274' 000000  000714*
4422 005275' 000000  005270'
4423 005276' 001000  000341             63300           POP     H               ;STILL POINTING TO THE FINAL VALUE
4424 005277' 001000  000301             63320           POP     B               ;GET THE SIGN  OF THE INCREMENT
4425 005300' 001000  000220             63340           SUB     B               ;SUBTRACT THE INCREMENTS SIGN FROM THAT
4426                                    63360                                   ;OF (CURRENT VALUE-FINAL VALUE)
4427 005301' 001000  000315             63380           CALL    MOVRM           ;GET LINE # OF "FOR" INTO (D,E)
4428 005302' 000000  005270'
4429 005303' 000000  005274'
4430                                    63400                                   ;GET TEXT POINTER OF "FOR" INTO (B,C)
4431 005304' 001000  000312             63420           JZ      LOOPDN          ;IF SIGN(FINAL-CURRENT)*SIGN(STEP)=0
4432 005305' 000000  005320'
4433 005306' 000000  005306'
4434                                    63440                                   ;THEN THE LOOP IS FINISHED
4435 005307' 001000  000353             63460           XCHG
4436 005310' 001000  000042             63480           SHLD    CURLIN          ;STORE THE LINE #
4437 005311' 000000  001607'
4438 005312' 000000  005305'
4439 005313' 001000  000151             63500           MOV     L,C             ;SETUP THE TEXT POINTER
4440 005314' 001000  000140             63520           MOV     H,B
4441 005315' 001000  000303             63540           JMP     NXTCON
4442 005316' 000000  003276'
4443 005317' 000000  005311'
4444
4445 005320' 001000  000371             63560   LOOPDN: SPHL                    ;ELIMINATE THE FOR ENTRY
4446                                    63600                                   ;SINCE [H,L] MOVED ALL
```

```
 4447                                   63620                                  ;THE WAY DOWN THE ENTRY
 4448  003321' 001000  000652           63640          LHLD    TEMP            ;RESTORE THE TEXT POINTER
 4449  003324' 000000  001603'
 4450  003327' 000000  003316'
 4451                                   63660  IFE     LENGTH,<JMP     NEWSTT>
 4452                                   63680  IFN     LENGTH,<
 4453  003324' 001000  000176           63700          MOV     A,M             ;IS THERE A COMMA AT THE END
 4454  003325' 001000  000376           63720          CPI     44              ;IF SO LOOK AT ANOTHER
 4455  003326' 000000  000054
 4456  003327' 001000  000302           63740          JNZ     NEWSTT          ;VARIABLE NAME TO "NEXT"
 4457  003330' 000000  003322'
 4458  003331' 000000  003322'
 4459  003332' 001000  000327           63760          CMMGET                  ;READ FIRST CHARCTER
 4460  003333' 001000  000315           63780          CALL    NEXTC>          ;DO NEXT, BUT DON'T ALLOW
 4461  003334' 000000  003230'
 4462  003335' 000000  003330'
 4463                                   63800                                  ;BLANK VARIABLE NAME (D,E)=STK PTR
 4464                                   63820                                  ;AND WILL NEVER MATCH ANY VARPTR
 4465                                   63840                                  ;USE CALL TO PUT DUMMY "NEWSTT" ENTRY ON
 4466
 4467                                   63860  PAGE
```



```
 4468
 4469
```

```
                                00020   SUBTTL FORMULA EVALUATION CODE
                                00040
                                00060   IFN     LENGTH-2,<
                                00080   IFN     STRING,<
                                00100   ;
                                00120   ; THESE ROUTINES CHECK FOR A CERTAIN VALTYP
                                00140   ; [A] IS NOT PRESERVED
                                00160   ;
                                00180   FRMNUM: CALL    FRMEVL          ;EVALUATE A FORMULA
                                00200   CHKNUM: XWD     "D1000,"D366    ;TURN CARRY OFF WITH ORI
                                00220   CHKSTR: STC                     ;SET CARRY
                                00240   CHKVAL: LDA     VALTYP          ;0 MEANS NUMERIC 1 MEANS STRING
                                00260           ADC     A               ;RESULT SHOULD BE 0 OR 3
                                00280                                   ;BAD RESULTS ARE 2 AND 1
                                00300           RPE                     ;RETURN IF CORRECT RESULT
                                00320   TMERR:  MVI     E,ERRTM         ;"TYPE MISMATCH ERROR"
                                00340           JMP     ERROR>>
                                00360   ;
                                00380   ; THE FORMULA EVALUATOR STARTS WITH
                                00400   ; [H,L] POINTING TO THE FIRST CHARACTER OF THE FORMULA,
                                00420   ; AT THE END [H,L] POINTS TO THE TERMINATOR,
                                00440   ; THE RESULT IS LEFT IN THE FAC,
                                00460   ;
                                00480   ; THE FORMULA EVALUATOR USES THE OPERATOR TABLE (OPTAB)
                                00500   ; TO DETERMINE PRECEDENCE AND DISPATCH ADDRESSES FOR
                                00520   ; EACH OPERATOR,
                                00540   ; A TEMPORARY RESULT ON THE STACK HAS THE FOLLOWING FORMAT
                                00560   ;
                                00580   ; THE ADDRESS OF 'RETADP' == THE PLACE TO RETURN ON COMPLETION
                                00600   ; OF OPERATOR APPLICATION
                                00620   ;
                                00640   ; THE FLOATING POINT TEMPORARY RESULT
                                00660   ;
                                00680   ; THE ADDRESS OF THE OPERATOR ROUTINE
                                00700   ;
                                00720   ; THE PRECEDENCE OF THE OPERATOR
                                00740   ;
                                00760   ; TOTAL 10 BYTES
                                00780   ;
                                00800   IFE     STRING,<FRMNUM=>
005336' 001000  000053          00820   FRMEVL: DCX     H               ;BACK UP CHARACTER POINTER
005337' 001000  000026          00840   FRMCHK: MVI     D,0             ;INITIAL DUMMY PRECEDENCE IS 0
005340' 000000  000000          00860           PUSH    D               ;SAVE PRECEDENCE
005341' 001000  000325          00880   LPOPEN: CALL    GETSTK          ;MAKE SURE THERE IS ROOM FOR RECURSIVE CALLS
005343' 000000  000024'
005344' 000000  005334'
005345' 000000  000001          00900           1
005346' 001000  000315          00920           CALL    EVAL            ;EVALUATE SOMETHING
005347' 000000  006061'
005350' 000000  005345'
005351' 001000  000042          00940           SHLD    TEMP2           ;SAVE TEXT POINTER
005352' 000000  001605'
```



```
005353' 000000  005347'
005354' 001000  000052          00960   RETADP: LHLD    TEMP2           ;RESTORE TEXT PTR
005355' 000000  001605'
005356' 000000  005352'
005357' 001000  000301          00980   TSTOP:  POP     B               ;POP OFF THE PRECEDENCE OF OLDOP
                                01000           IFN     LENGTH-2,<
                                01020           IFN     STRING,<
                                01040           MOV     A,B             ;LOOK AT PRECEDENCE
                                01060           CPI     120             ;IF IT'S SOME ARITHMETIC
                                01080                                   ;(TYPE THING WE SHOULDN'T
                                01100                                   ;SEE STRINGS
                                01120           CNC     CHKNUM>>        ;TYPE MISMATCH ERROR IF NOT NUMERIC
005360' 001000  000176          01140   NOTSTV: MOV     A,M             ;GET NEXT CHARACTER
                                01160           IFN     LENGTH,<
005361' 001000  000026          01180           MVI     D,0             ;ASSUME NO RELATION OPS
005362' 000000  000000
005363' 001000  000328          01200   LUPREL: SUI     GREATK          ;IS THIS ONE RELATION?
005364' 000000  000257
005365' 001000  000332          01220           JC      ENDREL          ;RELATIONS ALL THROUGH
005366' 000000  005415'
                                01240   NMREL=-LESSTK-GREATK+1
005367' 000000  005355'
                                000003
005370' 001000  000376          01260           CPI     NMREL           ;IS IT REALLY RELATIONAL?
005371' 000000  000003
005372' 001000  000322          01280           JNC     ENDREL          ;NO JUST BIG
005373' 000000  005415'
005374' 000000  005366'
005375' 001000  000376          01300           CPI     1               ;SET UP BITS BY MAPPING
005376' 000000  000001
005377' 001000  000027          01320           RAL                     ;0 TO 1 1 TO 2 AND 2 TO 4
005400' 001000  000252          01340           XRA     D               ;BRING IN THE OLD BITS
005401' 001000  000272          01360           CMP     D               ;MAKE SURE RESULT IS BIGGER
005402' 001000  000127          01380           MOV     D,A             ;SAVE THE MASK
005403' 001000  000332          01400           JC      SNERR           ;DON'T ALLOW TWO OF THE SAME
005404' 000000  000272'
005405' 000000  005373'
005406' 001000  000042          01420           SHLD    TEMP3           ;SAVE CHARACTER POINTER
005407' 000000  001575'
005410' 000000  005404'
005411' 001000  000327          01440           CHRGET                  ;GET THE NEXT CANDIDATE
005412' 001000  000303          01460           JMP     LUPREL
005413' 000000  005363'
005414' 000000  005407'
005415' 001000  000172          01480   ENDREL: MOV     A,D             ;GET THE MASK
005416' 001000  000260          01500           ORA     A               ;WERE THERE ANY?
005417' 001000  000302          01520           JNZ     FINREL          ;IF SO, HANDLE AS SPECIAL OP
005420' 000000  005603'
005421' 000000  005413'
005422' 001000  000176          01540           MOV     A,M             ;GET THE CHARACTER AGAIN
005423' 001000  000042          01560           SHLD    TEMP3>          ;SAVE UPDATED CHARACTER POINTER
005424' 000000  001575'
005425' 000000  005420'
005426' 001000  000326          01580           SUI     PLUSTK          ;AN OPERATOR?
```

```
                                01600           RC                      ;RETURN IF NOT
                                01620                                   ;THIS CAN RESULT IN OPERATOR
                                01640                                   ;APPLICATION OR ACTUAL RETURN
4580  005431' 001000  000376    01660           CPI     LSTOPK          ;HIGHER THAN THE LAST OP?
4581  005433' 000000  000007
4582  005433' 001000  000320    01680           RNC
4583  005434' 001000  000137    01700           MOV     E,A             ;MUST MULTIPLY BY 3 SINCE
4584                            01720                                   ;OPTAB ENTRIES ARE 3 LONG
4585                            01740   IFN     STRING,<
4586  005435' 001000  000072    01760           LDA     VALTYP          ;SEE IF LEFT PART IS STRING
4587  005436' 000000  001542'
4588  005437' 000000  005424'
4589                            01780   IFE     LENGTH=2,<
4590  005440' 001000  000376    01800           CPI     3>              ;SEE IF ITS A STRING
4591  005441' 000000  000003
4592                            01820   IFN     LENGTH=2,<
4593                            01840           ORA     A>
4594  005442' 001000  000263    01860           ORA     E               ;SET CONDITION CODES
4595                            01880   IFN     LENGTH=2,<
4596                            01900           MOV     A,E>            ;REFETCH OP-VALUE
4597  005443' 001000  000312    01920           JZ      CAT>            ;MUST BE CAT
4598  005444' 000000  010320'
4599  005445' 000000  005436'
4600                            01940   IFN     LENGTH=2,<
4601                            01960           RLC                     ;A= ORIGINAL A*2
4602                            01980           ADD     E               ;ADD IN ORIGINAL A
4603                            02000           MOV     E,A>            ;CREATE TWO BYTE VALUE
4604                            02020   IFE     LENGTH,<
4605                            02040           MVI     D,0>            ;HIGH ORDER =0
4606  005446' 001000  000041    02060           LXI     H,OPTAB         ;CREATE INDEX INTO OPTAB
4607  005447' 000000  000163'
4608  005450' 000000  005444'
4609  005451' 001000  000031    02080           DAD     D               ;ADD IN CALCULATED OFFSET
4610  005452' 001000  000170    02100           MOV     A,B             ;(A) GETS OLD PRECEDENCE
4611  005453' 001000  000120    02120           MOV     D,M             ;REMEMBER NEW PRECEDENCE
4612  005454' 001000  000272    02140           CMP     D               ;OLD-NEW
4613  005455' 001000  000320    02160           RNC                     ;MUST APPLY OLD OP
4614                            02180                                   ;IF HAS GREATER OR = PRECEDENCE
4615                            02200   IFN     LENGTH=2,<
4616                            02220           INX     H               ;NOW POINTING AT ROUTINE ADDRESS
4617                            02240   IFN     STRING,<CALL    CHKNUM> ;CAN'T BE STRING HERE
4618                            02260                                   ;SINCE THE ONLY STRING OPERATOR
4619                            02280                                   ;IS PLUS AND RELATIONALS
4620                            02300                                   ;DON'T COME THROUGH HERE
4621                            02320   DUPREC: PUSH    B               ;SAVE OLD PRECEDENCE
4622                            02340           LXI     B,RETAOP        ;OPERATOR RETURN ADDRESS
4623                            02360           PUSH    B               ;FIRST PART OF "TEMP" ENTRY
4624                            02380   IFN     LENGTH,<
4625                            02400           MOV     B,E>            ;SAVE SECOND BYTE OF PRECEDENCE
4626                            02420                                   ;SINCE FOR RELATIONAL OPERATORS
4627                            02440                                   ;IT GIVES THE VALUE TYPE OF
4628                            02460                                   ;THE LEFT SIDE AND IT TELLS
```



```
                                02480                                   ;WHICH RELATIONAL-OPERATOR
                                02500                                   ;IT WAS
                                02520           MOV     C,0
                                02540           CALL    PUSHF
                                02560   IFN     LENGTH,<
                                02580           MOV     E,B>            ;GET SECOND BYTE OF PRECEDENCE AGAIN
                                02600           MOV     D,C             ;(D) GETS PRECEDENCE
                                02620           PUSHM                   ;PUT ROUTINE ADDRESS ON THE STACK
                                02640   IFN     LENGTH,<
                                02660           LHLD    TEMP3>
                                02680   IFE     LENGTH,<
                                02700           LHLD    TEMP2>          ;IF WE DONT HAVE "LENGTH"
                                02720                                   ;ON OPERATORS CAN ONLY
                                02740                                   ;BE ONE CHARACTER SO TO "REREAD"
                                02760                                   ;AN OPERATOR THAT WE LOOKED
                                02780                                   ;AT BEFORE AND DECIDED NOT TO
                                02800                                   ;APPLY WE JUST "DCX H"
                                02820                                   ;IF LENGTH IS ON WE HAVE TO
                                02840                                   ;REMEMBER THE TEXT POINTER BEFORE
                                02860                                   ;THE OPERATOR AND AFTER SO WE CAN
                                02880                                   ;EITHER RESCAN THE OPERATOR
                                02900                                   ;LATER IF IT DOESN'T GET APPLIED
                                02920                                   ;OR GO BEYOND IT WHEN IT DOES
                                02940           JMP     LPOPER>         ;PUT ON PRECEDENCE AND LOOK AT A
                                02960                                   ;NEW OPERATOR
4655                            03000   IFE     LENGTH=2,<
4656  005456' 001000  000305    03020           PUSH    B               ;SAVE THE OLD PRECEDENCE
4657  005457' 001000  000001    03040           LXI     B,RETAOP        ;PUT ON THE ADDRESS OF THE
4658  005460' 000000  005354'
4659  005461' 000000  005447'
4660                            03060           PUSH    B               ;PLACE TO RETURN TO AFTER OPERATOR APPLICATION
4661  005462' 001007  000305
4662  005463' 001000  000172    03080           MOV     A,D             ;SEE IF THE OPERATOR IS EXPONENTIATION
4663  005464' 001000  000376    03100           CPI     12?             ;WHICH HAS PRECEDENCE 12?
4664  005465' 000000  000177
4665                            03120           JZ      EXPSTK          ;IF SO, "FRCSNG" AND MAKE A SPECIAL STACK ENTRY
4666  005466' 001000  000312
4667  005467' 000000  005553'
4668  005470' 000000  005466'
4669  005471' 001000  000376    03140           CPI     81              ;SEE IF THE OPERATOR IS "AND" OR "OR"
4670  005472' 000000  000121
4671  005473' 001000  000332    03160           JC      ANJORD          ;AND IF SO "FRCINT" AND
4672  005474' 000000  005571'
4673  005475' 000000  005467'
4674                            03180                                   ;MAKE A SPECIAL STACK ENTRY
4675                            03200   ;
4676                            03220   ; THIS CODE PUSHES THE CURRENT VALUE IN THE FAC
4677                            03240   ; ONTO THE STACK, EXCEPT IN THE CASE OF STRINGS IN WHICH IT CALLS
4678                            03260   ; TYPE MISMATCH ERROR. (D) AND (E) ARE PRESERVED.
4679                            03280   ;
4680  005476' 001000  000072    03300   NUMPRE: LDA     VALTYP          ;GET THE VALUE TYPE
4681  005477' 000000  001543'
```

```
4682  005500' 000000  005474'
4683  005501' 001000  000376          03300          CPI     3               ;AND SET THE CONDITION CODES BASED ON IT
4684  005502' 000000  000003
4685  005503' 001000  000312          03320          JZ      TMERR           ;BLOW UP ON STRINGS
4686  005504' 000000  0020A0*
4687  005505' 000000  005477'
4688  005506' 00100F  000041          03340          LXI     H,FACLO         ;GET POINTER TO LO IN FAC
4689  005507' 000000  001637'
4690  005510' 000000  005504'
4691  005511' 001000  000116          03342          MOV     C,M
4692  005512' 001000  000043          03344          INX     H
4693  005513' 001000  000106          03346          MOV     B,M
4694  005514' 001000  000043          03348          INX     H
4695  005515' 001000  000305          03360          PUSH    B               ;PUSH FACLO*8,1 ON THE STACK
4696  005516' 001000  000372          03380          JM      VPUSHD          ;ALL DONE IF THE DATA WAS AN INTEGER
4697  005517' 000000  005536'
4698  005520' 000000  005507'
4699  005521' 001000  000116          03382          MOV     C,M
4700  005522' 001000  000043          03384          INX     H
4701  005523' 001000  000106          03386          MOV     B,M
4702  005524' 001000  000043          03388          INX     H
4703  005525' 001000  000305          03400          PUSH    B               ;PUSH FAC*1,0 ON THE STACK
4704  005526' 001000  202342          03420          JPO     VPUSHD          ;ALL DONE IF WE HAD A SNG
4705  005527' 000000  005536'
4706  005530' 000000  005517'
4707  005531' 001000  000041          03440          LXI     H,DFACLO        ;WE HAVE A DOUBLE PRECISON NUMBER
4708  005532' 000000  001634'
4709  005533' 000000  005527'
4710  005534' 001000  000367          03460          PUSHM                   ;PUSH ITS 4 LO BYTES ON THE STACK
4711  005535' 001000  000367          03480          PUSHM
4712  005536' 001000  000113          03520  VPUSHD: MOV     C,E             ;[C]=OPERATOR NUMBER
4713  005537' 001000  000107          03540          MOV     B,A             ;[B]=TYPE OF VALUE ON THE STACK
4714  005540' 001000  000305          03560          PUSH    B               ;SAVE THESE THINGS FOR APPLOP
4715  005541' 001000  000001          03600          LXI     B,APPLOP        ;GENERAL OPERATOR APPLICATION
4716  005542' 000000  005642'
4717  005543' 000000  005536'
4718                                   03620                                  ;ROUTINE WE DOES TYPE CONVERSIONS
4719  005544' 001000  000305          03640  FINTMP: PUSH    B               ;SAVE PLACE TO GO
4720  005545' 001000  000052          03660          LHLD    TEMP3           ;REGET THE TEXT POINTER
4721  005546' 000000  001575'
4722  005547' 000000  005546'
4723  005550' 001000  000303          03680          JMP     LPOPER          ;PUSH ON THE PRECEDENCE AND READ MORE
4724  005551' 000000  005341'
4725  005552' 000000  005546'
4726                                   03700                                  ;FORMULA
4727                                   03720  ;
4728                                   03740  ; FOR EXPONENTIATION WE WANT TO FORCE THE CURRENT VALUE IN THE FAC
4729                                   03760  ; TO BE SINGLE PRECISION, WHEN APPLICATION TIME COMES WE FORCE
4730                                   03780  ; THE RIGHT HAND OPERAND TO SINGLE PRECISION AS WELL
4731                                   03800  ;
4732  005553' 001000  000315          03820  EXPSTK: CALL    FRCSNG          ;COERCE LEFT HAND OPERAND
4733  005554' 000000  003C56*
4734  005555' 000000  005551'
```



```
4735  005556' 001000  000315          03840          CALL    PUSHF           ;PUT IT ON THE STACK
4736  005557' 000000  0000204
4737  005560' 000000  005554'
4738  005561' 001000  000001          03860          LXI     B,FPWRRR        ;PLACE TO COERCE RIGHT HAND
4739  005562' 000000  0000004
4740  005563' 000000  005557'
4741                                   03880                                  ;OPERAND AND DO EXPONENTIATION
4742  005564' 001000  000026          03900          MVI     D,127           ;RESTORE THE PRECEDENCE
4743  005565' 000000  000177
4744  005566' 001000  000303          03920          JMP     FINTMP          ;FINISH ENTRY AND EVALUATE MORE FORMULA
4745  005567' 000000  005544'
4746  005570' 000000  005562'
4747                                   03940  ;
4748                                   03960  ; FOR "AND" AND "OR" WE WANT TO FORCE THE CURRENT VALUE IN THE
4749                                   03980  ; FAC TO BE AN INTEGER, AND AT APPLICATION TIME FORCE THE RIGHT
4750                                   04000  ; HAND OPERAND TO BE AN INTEGER
4751                                   04020  ;
4752  005571' 001000  000325          04040  ANDORD: PUSH    D               ;SAVE THE PRECEDENCE (70 OR 80)
4753  005572' 001000  000315          04060          CALL    FRCINT
4754  005573' 000000  003030*
4755  005574' 000000  005567'
4756  005575' 001000  000321          04080          POP     D               ;[D]=PRECEDENCE
4757  005576' 001000  000345          04100          PUSH    H               ;PUSH THE LEFT HAND OPERAND
4758  005577' 001000  000021          04120          LXI     D,OANDOR        ;"AND" AND "OR" ODER
4759  005600' 000000  0004037'
4760  005601' 000000  005573'
4761  005602' 001000  000303          04140          JMP     FINTMP          ;PUSH ON THIS ADDRESS,PRECEDENCE
4762  005603' 000000  005544'
4763  005604' 000000  005600'
4764                                   04160                                  ;AND CONTINUE EVALUATION
4765                                   04180  ;
4766                                   04200  ; HERE TO BUILD AN ENTRY FOR A RELATIONAL OPERATOR
4767                                   04220  ; STRINGS ARE TREATED SPECIALLY, NUMERIC COMPARES ARE DIFFERENT
4768                                   04240  ; FROM MOST OPERATOR ENTRIES ONLY IN THE FACT THAT AT THE
4769                                   04260  ; BOTTOM INSTEAD OF HAVING RETAOP, DOCMP AND THE RELATIONAL
4770                                   04280  ; BITS ARE STORED, STRINGS HAVE STRCMP,THE POINTER AT THE STRING DESCRIPTOR,
4771                                   04300  ; DOCMP AND THE RELATIONAL BITS.
4772                                   04320  ;
4773  005605' 001000  000170          04340  FINREL: MOV     A,B             ;[A]=OLD PRECEDENCE
4774  005606' 001000  000376          04360          CPI     100             ;RELATIONALS HAVE PRECEDENCE 100
4775  005607' 000000  000144
4776  005610' 001000  000320          04380          RNC                     ;APPLY EARLIER OPERATOR IF IT HAS
4777                                   04400                                  ;HIGHER PRECEDENCE
4778  005611' 001000  000305          04420          PUSH    B               ;SAVE THE OLD PRECEDENCE
4779  005612' 001000  000325          04440          PUSH    D               ;SAVE [D]=RELATIONAL BITS
4780  005613' 001000  000021          04460          LXI     D,$CODE+25604   ;[D]=PRECEDENCE=100
4781  005614' 000000  062004'
4782  005615' 000000  005603'
4783                                   04480                                  ;[E]=DISPATCH OFFSET FOR
4784                                   04500                                  ;COMPARES IN APPLOP=4
4785                                   04520                                  ;IN CASE THIS IS A NUMERIC COMPARE
4786  005616' 001000  000041          04540          LXI     H,DOCMP         ;ROUTINE TO TAKE COMPARE ROUTINE RESULT
4787  005617' 000000  006373'
```

```
 4788  005620' 000000  005614'
 4789                                  04560                                   ;AND RELATIONAL BITS AND RETURN THE ANSWER
 4790  005621' 021000  000345         04580           PUSH    H               ;DOES A JMP TO RETADP WHEN DONE
 4791  005622' 021000  000315         04600           CALL    GETYPE          ;SEE IF WE HAVE A NUMERIC COMPARE
 4792  005625' 000000  006407'
 4793  005624' 000000  005617'
 4794  005625' 021000  000302         04620           JNZ     NUMREL          ;YES, BUILD AN APPLOP ENTRY
 4795  005626' 000000  005476'
 4796  005627' 000000  005623'
 4797  005630' 021000  000052         04640           LHLD    FACLO           ;GET THE POINTER AT THE STRING DESCRIPTOR
 4798  005631' 000000  001637'
 4799  005632' 000000  005626'
 4800  005633' 021000  000345         04660           PUSH    H               ;SAVE IT FOR STRCMP
 4801  005634' 021000  000001         04680           LXI     B,STRCMP        ;STRING COMPARE ROUTINE
 4802  005635' 000000  016320'
 4803  005636' 000000  005631'
 4804  005637' 021000  000303         04700           JMP     FINTMP          ;PUSH THE ADDRESS, HEGET THE TEXT POINTER
 4805  005640' 000000  005544'
 4806  005641' 000000  005615'
 4807                                  04720                                   ;SAVE THE PRECEDENCE AND SCAN
 4808                                  04722                                   ;MORE OF THE FORMULA
 4809                                  04740   }
 4810                                  04760   ; APPLOP IS RETURNED TO WHEN IT IS TIME TO APPLY AN ARITHMETIC
 4811                                  04762   ; OR NUMERIC COMPARISON OPERATION.
 4812                                  04764   ; THE STACK HAS A DOUBLE BYTE ENTRY WITH THE OPERATOR
 4813                                  04766   ; NUMBER AND THE VALTYP OF THE VALUE ON THE STACK.
 4814                                  04768   ; APPLOP DECIDES WHAT VALUE LEVEL THE OPERATION
 4815                                  04770   ; WILL OCCUR AT, AND CONVERTS THE ARGUMENTS. APPLOP
 4816                                  04772   ; USES DIFFERENT CALLING CONVENTIONS FOR EACH VALUE TYPE.
 4817                                  04776   ; INTEGERS: LEFT IN [D,E] RIGHT IN [H,L]
 4818                                  04778   ; SINGLES:  LEFT IN [B,C,D,E] RIGHT IN THE FAC
 4819                                  04780   ; DOUBLES:  LEFT IN FAC   RIGHT IN ARG
 4820                                  04798   }
 4821  005642' 001000  000301         04800   APPLOP: POP     B               ;[B]=STACK OPERAND VALUE TYPE
 4822                                  04820                                   ;[C]=OPERATOR OFFSET
 4823  005643' 001000  000171         04840           MOV     A,C             ;SAVE IN MEMORY SINCE THE STACK WILL BE BUSY
 4824  005644' 001000  000062         04860           STA     OPRTYP          ;A RAM LOCATION
 4825  005645' 000000  001544'
 4826  005646' 000000  005640'
 4827  005647' 001000  000170         04880           MOV     A,B             ;CHECK FOR DOUBLE
 4828  005650' 001000  000376         04900           CPI     8               ;PRECISION ENTRY ON THE STACK
 4829  005651' 000000  000010
 4830  005652' 001000  000312         04920           JZ      STKDBL          ;FORCE FAC TO DOUBLE
 4831  005653' 000000  005725'
 4832  005654' 000000  005645'
 4833  005655' 001000  000072         04940           LDA     VALTYP          ;SEE IF THE FAC IS DOUBLE PRECISION
 4834  005656' 000000  001543'
 4835  005657' 000000  005653'
 4836  005660' 001000  000376         04960           CPI     8               ;AND IF SO, CONVERT THE STACK OPERAND
 4837  005661' 000000  000010
 4838  005662' 001000  000312         04980           JZ      FACDBL          ;TO DOUBLE PRECISION
 4839  005663' 000000  005775'
 4840  005664' 000000  005656'
```



```
 4841  005665' 001000  000127         05000           MOV     D,A             ;SAVE THE VALUE TYPE OF THE FAC
 4842  005666' 021000  000170         05020           MOV     A,B             ;SEE IF THE STACK ENTRY IS SINGLE
 4843  005667' 001000  000376         05040           CPI     4               ;PRECISION AND IF SO, CONVERT
 4844  005670' 000000  000004
 4845  005671' 001000  000312         05060           JZ      STKSNG          ;THE FAC TO SINGLE PRECISION
 4846  005672' 000000  005621'
 4847  005673' 000000  005663'
 4848  005674' 001000  000170         05080           MOV     A,B             ;SEE IF THE FAC IS SINGLE PRECISION
 4849  005675' 001000  000376         05100           CPI     3               ;AND IF SO CONVERT THE STACK TO SINGLE
 4850  005676' 000000  000003
 4851  005677' 001000  000322         05120           JNC     FACSNG          ;PRECISION
 4852  005700' 000000  000034'
 4853  005701' 000000  005672'
 4854                                  05140                                   ;NOTE: THE STACK MUST BE INTEGER AT THIS POINT
 4855  005702' 001000  000312         05160           JZ      TMERR           ;BLOW UP ON RIGHT HAND STRING OPERAND
 4856  005703' 000000  005504A
 4857  005704' 000000  005700'
 4858  005705' 001000  000041         05180           LXI     H,INTOSP        ;INTEGER INTEGER CASE
 4859  005706' 000000  000716'
 4860  005707' 000000  005703'
 4861  005710' 001000  000006         05200           MVI     B,0             ;SPECIAL DISPATCH FOR SPEED
 4862  005711' 000000  000000
 4863  005712' 001000  000011         05220           DAD     B               ;[H,L] POINTS TO THE ADDRESS TO GO TO
 4864  005713' 001000  000011         05240           DAD     B
 4865  005714' 001000  000116         05260           MOV     C,M             ;[B,C]=ROUTINE ADDRESS
 4866  005715' 001000  000143         05280           INX     H
 4867  005716' 001000  000106         05300           MOV     B,M
 4868  005717' 001000  000321         05320           POP     D               ;[D,E]=LEFT HAND OPERAND
 4869  005720' 001000  000052         05340           LHLD    FACLO           ;[H,L]=RIGHT HAND OPERAND
 4870  005721' 000000  001637'
 4871  005722' 000000  005706'
 4872  005725' 001000  000305         05360           PUSH    B               ;DISPATCH
 4873  005724' 001000  000311         05380           RET
 4874                                  05400   }
 4875                                  05420   ; THE STACK OPERAND IS DOUBLE PRECISION, SO
 4876                                  05440   ; THE FAC MUST BE FORCED TO DOUBLE PRECISION, MOVED INTO ARG
 4877                                  05460   ; AND THE STACK VALUE POPED INTO THE FAC
 4878                                  05480   }
 4879  005725' 001000  000315         05500   STKDBL: CALL    FMDUBL          ;MAKE THE FAC DOUBLE PRECISION
 4880  005726' 000000  000044
 4881  005727' 000000  005721'
 4882  005730' 001000  000315         05520           CALL    VMOVAF          ;MOVE THE FAC INTO ARG
 4883  005731' 000000  000000*
 4884  005732' 000000  005726'
 4885  005735' 001000  000341         05540           POP     H               ;POP OFF THE STACK OPERAND INTO THE FAC
 4886  005734' 001000  000042         05560           SHLD    OPACLO+2
 4887  005735' 000000  001635'
 4888  005736' 000000  005731'
 4889  005737' 001000  000341         05580           POP     H
 4890  005740' 001000  000042         05600           SHLD    OPACLO          ;STORE LOW BYTES AWAY
 4891  005741' 000000  001633'
 4892  005742' 000000  005735'
 4893  005745' 001000  000301         05620   SNGDBL: POPR                    ;POP OFF A FOUR BYTE VALUE
```

```
 4894  005744' 001000  000321
 4895  005745' 001000  000315          05640           CALL    MOVFR           ;INTO THE FAC
 4896  005746' 000000  000000*
 4897  005747' 000000  005741'
 4898  005750' 001000  000315          05660   SETDBL: CALL    FRCDBL          ;MAKE SURE THE LEFT OPERAND IS
 4899  005751' 000000  005726*
 4900  005752' 000000  005746'
 4901                                   05680                                   ;DOUBLE PRECISION
 4902  005753' 001000  000041          05700           LXI     H,DBLDSP        ;DISPATCH TO A DOUBLE PRECISION ROUTINE
 4903  005754' 000000  006672'
 4904  005755' 000000  005751'
 4905  005756' 001000  000072          05720   DODSP:  LDA     OPRTYP          ;RECALL WHICH OPERAND IT WAS
 4906  005757' 000000  001544'
 4907  005760' 000000  005754'
 4908  005761' 001000  000007          05740           RLC                     ;CREATE A DISPATCH OFFSET, SINCE
 4909                                   05760                                   ;TABLE ADDRESSES ARE TWO BYTES
 4910  005762' 001000  000305          05780           PUSH    B               ;SAVE (B,C) FOR SINGLE PRECISION
 4911  005763' 001000  000117          05800           MOV     C,A             ;DOUBLE BYTE OFFSET
 4912  005764' 001000  000000          05820           MVI     B,0             ;INTO (B,C)
 4913  005765' 000000  000000
 4914  005766' 001000  000011          05840           DAD     B               ;CALCULATE LOCATION OF ROUTINE TO GO TO
 4915  005767' 001000  000301          05860           POP     B               ;GET BACK (B,C) FOR SINGLE PRECISION
 4916  005770' 001000  000176          05880           MOV     A,M             ;GET THE ADDRESS
 4917  005771' 001000  000043          05900           INX     H
 4918  005772' 001000  000146          05920           MOV     H,M
 4919  005773' 001000  000157          05940           MOV     L,A
 4920  005774' 001000  000351          05960           PCHL                    ;AND PERFORM THE OPERATION, RETURNING
 4921                                   05980                                   ;TO RETADR, EXCEPT FOR COMPARES WHICH
 4922                                   06000                                   ;RETURN TO DOCMP
 4923                                   06020   ;
 4924                                   06040   ; THE FAC IS DOUBLE PRECISION AND THE STACK IS EITHER
 4925                                   06060   ; INTEGER OR SINGLE PRECISION AND MUST BE CONVERTED
 4926                                   06080   ;
 4927  005775' 001000  000305          06100   FACDBL: PUSH    B               ;SAVE THE STACK VALUE TYPE
 4928  005776' 001000  000315          06120           CALL    MOVAF           ;MOVE THE FAC INTO ARG
 4929  005777' 000000  005731*
 4930  006000' 000000  005757'
 4931  006001' 001000  000361          06140           POP     PSW             ;POP THE STACK VALUE TYPE INTO [A]
 4932  006002' 001000  000062          06160           STA     VALTYP          ;PUT IT IN VALTYP FOR THE FORCE
 4933  006003' 000000  001543'
 4934  006004' 000000  005777'
 4935                                   06180                                   ;ROUTINE
 4936  006005' 001000  000376          06200           CPI     4               ;SEE IF ITS SINGLE, SO WE KNOW
 4937  006006' 000002  000004
 4938                                   06220                                   ;HOW TO POP THE VALUE OFF
 4939  006007' 001000  000312          06240           JZ      SNGDBL          ;IT'S SINGLE PRECISION
 4940  006010' 000000  005743'
 4941  006011' 000000  006605'
 4942                                   06260                                   ;SO DO A POPR / CALL MOVFR
 4943  006012' 001000  000341          06280           POP     H               ;POP OFF THE INTEGER VALUE
 4944  006013' 001000  000042          06300           SHLD    FACLO           ;SAVE IT FOR CONVERSION
 4945  006014' 000000  001637'
 4946  006015' 000000  006610'
```



```
 4947  006016' 001000  000303          06320           JMP     SETDBL          ;SET IT UP
 4948  006017' 000000  005754'
 4949  006020' 000000  006614'
 4950                                   06340   ;
 4951                                   06360   ; THIS IS THE CASE WHERE THE STACK IS SINGLE PRECISION
 4952                                   06380   ; AND THE FAC IS EITHER SINGLE PRECISION OR INTEGER
 4953                                   06400   ;
 4954  006021' 001000  000315          06420   STKSNG: CALL    FRCSNG          ;CONVERT THE FAC IF NECESSARY
 4955  006022' 000000  005554*
 4956  006023' 000000  006017'
 4957  006024' 001000  000301          06440           POPR                    ;PUT THE LEFT HAND OPERAND IN THE REGISTERS
 4958  006025' 001000  000321
 4959  006026' 001000  000041          06460   SNGOO:  LXI     H,SNGDSP        ;SETUP THE DISPATCH ADDRESS
 4960  006027' 000000  000704'
 4961  006030' 000000  006622'
 4962                                   06480                                   ;FOR THE SINGLE PRECISION OPERATOR ROUTINES
 4963  006031' 001000  000303          06500           JMP     DODSP           ;DISPATCH
 4964  006032' 000000  005756'
 4965  006033' 000000  006027'
 4966                                   06520   ;
 4967                                   06540   ; THIS IS THE CASE WHERE THE FAC IS SINGLE PRECISION AND THE STACK
 4968                                   06560   ; IS AN INTEGER.
 4969                                   06580   ;
 4970  006034' 001000  000341          06600   FACSNG: POP     H               ;POP OFF THE INTEGER ON THE STACK
 4971  006035' 001000  000315          06620           CALL    PUSHF           ;SAVE THE FAC ON THE STACK
 4972  006036' 000000  005557*
 4973  006037' 000000  006032'
 4974  006040' 001000  000315          06640           CALL    CONSIM          ;CONVERT [H,L] TO A SINGLE PRECISION
 4975  006041' 000000  000000*
 4976  006042' 000000  006036'
 4977                                   06660                                   ;NUMBER IN THE FAC
 4978  006043' 001000  000315          06680           CALL    MOVFR           ;PUT THE LEFT HAND OPERAND IN THE REGISTERS
 4979  006044' 000000  003661*
 4980  006045' 000000  006041'
 4981  006046' 001000  000341          06700           POP     H               ;RESTORE THE FAC
 4982  006047' 001000  000042          06720           SHLD    FAC=1           ;FROM THE STACK
 4983  006050' 000000  001641'
 4984  006051' 000000  006044'
 4985  006052' 001000  000341          06740           POP     H
 4986  006053' 001000  000042          06760           SHLD    FACLO
 4987  006054' 000000  001637'
 4988  006055' 000000  006050'
 4989  006056' 001000  000303          06780           JMP     SNGOO>          ;PERFORM THE OPERATION
 4990  006057' 000000  006026'
 4991  006060' 000000  006054'
 4992
 4993  006061'                         06820   EVAL:
 4994                                   06840   IFN     LENGTH-2,<
 4995                                   06860   IFN     STRING,<
 4996                                   06880           XRA     A
 4997                                   06900           STA     VALTYP>>        ;ASSUME THE VALUE WILL BE NUMERIC
 4998  006061' 001000  000327          06920           CHRGET
 4999  006062' 001000  000332          06940           JC      FIN             ;IF NUMERIC, INTERPRET CONSTANT
```

```
                                                        ;VARIABLE NAME?
                        CALL    ISLET
                                                        ;AN ALPHABETIC CHARACTER MEANS YES
                        JNC     ISVAR
                                                        ;IGNORE "+"
                        CPI     PLUSTK
                        JZ      EVAL
                                                        ;"."  AS LEADING CHARACTER OF A
                        CPI     "."
                                                        ;CONSTANT?
                        JZ      FIN
                                                        ;NEGATION?
                        CPI     MINUTK
                        JZ      DOMIN
                                                        ;STRING CONSTANT?
                        CPI     34
                        JZ      STRLTI>                 ;IF SO BUILD A DESCRIPTOR IN A TEMPORARY
                                                        ;DESCRIPTOR LOCATION AND PUT A POINTER TO THE
                                                        ;DESCRIPTOR IN FACLO.
                        CPI     "N"                     ;CHECK FOR "NOT" OPERATOR
                        JZ      NOTER>
                        CPI     "F"                     ;USER-DEFINED FUNCTION?
                        JZ      FNDOER>
                        SUI     ONEFUN                  ;IS IT A FUNCTION NAME?
                        JNC     ISFUN                   ;FUNCTIONS ARE THE HIGHEST
                                                        ;NUMBERED CHARACTERS ALLOWED
                                                        ;SO THERE IS NO NEED TO CHECK
                                                        ;THE UPPER BOUND
PARCHK: SYNCHK  "("                                     ;ONLY POSSIBILITY LEFT
```



```
                                                        ;IS A FORMULA IN PARENTHESES
                        CALL    FRMEVL                  ;RECURSIVELY EVALUATE THE FORMULA
                        SYNCHK  ")"
                        RET
DOMIN:
                        IFE     EXTFNC,<
                        CALL    EVAL>                   ;NO "-" OPERATOR IN THIS CASE
                        IFN     EXTFNC,<
                        MVI     D,125                   ;A PRECEDENCE BELOW "-"
                                                        ;BUT ABOVE ALL ELSE
                        CALL    LPOPEN                  ;SO "-" GREATER THAN UNARY MINUS
                        LHLD    TEMP2>                  ;GET TEXT POINTER
                        PUSH    H
                        IFE     LENGTH-2,<
                        CALL    VNEG>
                        IFN     LENGTH-2,<
                        CALL    NEG>                    ;NEGATE THE FAC.
LABBCK:                                                 ;FUNCTIONS THAT DON'T RETURN
                                                        ;STRING VALUES COME BACK HERE
                        IFN     LENGTH-2,<
                        IFN     STRING,<
                        CALL    CHKNUM>>
                        POP     H
                        RET
ISVAR:  CALL    PTRGET                                  ;GET A POINTER TO THE
                                                        ;VARIABLE IN (D,E)
                        PUSH    H                       ;SAVE THE TEXT POINTER
                        IFE     STRING,<
                        XCHG                            ;TRANSFER THE POINTER AT THE VALUE
                                                        ;INTO (H,L)
                        CALL    MOVFM>                  ;SETUP FAC WITH VARIABLE VALUE
                        IFN     STRING,<
                        XCHG                            ;PUT THE POINTER TO THE VARIABLE VALUE
                                                        ;INTO (H,L). IN THE CASE OF A STRING
                                                        ;THIS IS A POINTER TO A DESCRIPTOR AND NOT
                                                        ;AN ACTUAL VALUE
                        SHLD    FACLO                   ;IN CASE IT'S STRING STORE THE POINTER
                                                        ;TO THE DESCRIPTOR IN FACLO.
```

```
                                 08240   IFE     LENGTH-2,<
 5107  006174' 001000  000315    08260           CALL    GETYPE          ;FOR STRINGS WE JUST LEAVE
 5108  006175' 000000  006307'
 5109  006176' 0F0000  006172'
 5110  006177' 001000  000304    08280           CNZ     VMOVFM>         ;A POINTER IN THE FAC
 5111  006200' 000000  0006004
 5112  006201' 000000  006175'
 5113                             08300   IFN     LENGTH-2,<
 5114                             08320           LDA     VALTYP
 5115                             08340           ORA     A
 5116                             08360           CZ      MOVFM>>         ;IF NOT,ACTUALLY TRANSFER THE VALUE INTO
 5117                             08380                                   ;THE FAC USING [H,L] AS THE POINTER,
 5118  006202' 001000  000341    08400           POP     H               ;RESTORE THE TEXT POINTER
 5119  006203' 001000  000311    08420           RET
 5120  006204' 001000  000006    08440   ISFUN:  MVI     B,0
 5121  006205' 000000  0D6000
 5122  006206' 001000  000007    08460           RLC                     ;MULTIPLY BY 2
 5123  006207' 001000  000117    08480           MOV     C,A
 5124  006210' 001000  000325    08500           PUSH    B               ;SAVE THE FUNCTION # ON THE STACK
 5125  006211' 001000  000327    08520           CMRGET
 5126                             08540   IFN     STRING,<
 5127  006212' 001000  000171    08560           MOV     A,C             ;LOOK AT FUNCTION #
 5128                             08580   NUMGFN==2*<LASNUM+2>+ONEFUN+1
 5129  006213' 001000  000376    08600           CPI     NUMGFN          ;IS IT PAST LASNUM?
 5130  006214' 000000  000051
 5131  006215' 001000  000332    08620           JC      OKNORM          ;NO,MUST BE NORMAL FUNCTION
 5132  006216' 000000  006251'
 5133  006217' 000000  006000'
 5134                             08640   ;
 5135                             08660   ; MOST FUNCTIONS TAKE A SINGLE ARGUMENT,
 5136                             08680   ; THE RETURN ADDRESS OF THESE FUNCTIONS IS A SMALL ROUTINE
 5137                             08700   ; THAT CHECKS TO MAKE SURE VALTYP IS 0 (NUMERIC) AND POPS OFF
 5138                             08720   ; THE TEXT POINTER, SO NORMAL FUNCTIONS THAT RETURN STRING RESULTS (I,E, CHR$)
 5139                             08740   ; MUST POP OFF THE RETURN ADDRESS OF LABBCK, AND POP OFF THE
 5140                             08760   ; TEXT POINTER AND THEN RETURN TO FRMEVL.
 5141                             08780   ;
 5142                             08800   ; THE SO CALLED "FUNNY" FUNCTIONS CAN TAKE MORE THAN ONE ARGUMENT,
 5143                             08820   ; THE FIRST OF WHICH MUST BE STRING AND THE SECOND OF WHICH
 5144                             08840   ; MUST BE A NUMBER BETWEEN 0 AND 256. THE TEXT POINTER IS
 5145                             08860   ; PASSED TO THESE FUNCTIONS SO ADDITIONAL ARGUMENTS
 5146                             08880   ; CAN BE READ. THE TEXT POINTER IS PASSED IN [D,E].
 5147                             08900   ; THE CLOSE PARENTHESIS MUST BE CHECKED AND RETURN IS DIRECTLY
 5148                             08920   ; TO FRMEVL WITH [H,L] SETUP AS THE TEXT POINTER POINTING BEYOND THE ")",
 5149                             08940   ; THE POINTER TO THE DESCRIPTION OF THE STRING ARGUMENT
 5150                             08960   ; IS STORED ON THE STACK UNDERNEATH THE VALUE OF THE INTEGER
 5151                             08980   ; ARGUMENT (2 BYTES)
 5152                             09000   ;
 5153  006220' 001000  000317    09020           SYNCHK  "("             ;FIRST ARGUMENT ALWAYS
 5154  006221' 000000  000050
 5155                             09040                                   ;STRING -- SECOND INTEGER
 5156  006222' 001000  000315    09060           CALL    FRMEVL          ;EAT OPEN PAREN AND FIRST ARG
 5157  006223' 000000  005336'
 5158  006224' 000000  006167'
```



```
 5159  006225' 001000  000317    09080           SYNCHK  44              ;TWO ARGS SO COMMA MUST DELIMIT
 5160  006226' 000000  0000054
 5161  006227' 001000  000315    09100           CALL    CHKSTR          ;MAKE SURE THE FIRST ONE WAS STRING
 5162  006230' 0F0000  004164*
 5163  006231' 000000  006223'
 5164  006232' 001000  000353    09120           XCHG                    ;[D,E]=TXTPTR
 5165  006233' 001000  000052    09140           LHLD    FACLO           ;GET PTR AT STRING DESCRIPTOR
 5166  006234' 000000  001637'
 5167  006235' 000000  006130'
 5168  006236' 001000  000343    09160           XTHL                    ;GET FUNCTION #
 5169                             09180                                   ;SAVE THE STRING PTR
 5170  006237' 001000  000345    09200           PUSH    H               ;PUT THE FUNCTION # ON
 5171  006240' 001000  000353    09220           XCHG                    ;[H,L]=TXTPTR
 5172  006241' 001000  000315    09240           CALL    GETBYT          ;[E]=VALUE OF FORMULA
 5173  006242' 000000  011020*
 5174  006243' 000000  006234'
 5175  006244' 001000  000353    09260           XCHG                    ;TEXT POINTER INTO [D,E]
 5176                             09280                                   ;[H,L]=INT VALUE OF SECOND ARGUMENT
 5177  006245' 001000  000345    09300           XTHL                    ;SAVE INT VALUE OF SECOND ARG
 5178                             09320                                   ;[H,L]=FUNCTION NUMBER
 5179  006246' 001000  000303    09340           JMP     FINGO>          ;DISPATCH TO FUNCTION
 5180  006247' 000000  006176'
 5181  006250' 000000  006242'
 5182  006251' 001000  000315    09360   OKNORM: CALL    PARCHK          ;MAKE SURE ITS THERE,
 5183  006252' 000000  006136'
 5184  006253' 000000  006247'
 5185                             09380                                   ;READ THE FORMULA INSIDE
 5186                             09400                                   ;AND MAKE SURE ITS FOLLOWED BY ")"
 5187  006254' 001000  000343    09420           XTHL                    ;[H,L]=FUNCTION # AND SAVE TEXT POINTER
 5188                             09440   IFE     LENGTH-2,<
 5189                             09460   ;
 5190                             09480   ; CHECK IF SPECIAL COERCION MUST BE DONE FOR ONE OF THE TRANSCENDENTAL
 5191                             09500   ; FUNCTIONS (RND, SQR, COS, SIN, TAN, ATN, LOG, AND EXP)
 5192                             09520   ;
 5193  006255' 001000  000175    09540           MOV     A,L             ;[A]=FUNCTION NUMBER
 5194                             000016  09550   BOTCON==<SQNTK-ONEFUN>*2
 5195  006256' 001000  000376    09560           CPI     BOTCON          ;LESS THAN SQUARE ROOT?
 5196  006257' 000000  000015
 5197  006260' 001000  000332    09580           JC      NOTFRF          ;DON'T FORCE THE ARGUMENT
 5198  006261' 000000  006272'
 5199  006262' 000000  006252'
 5200                             000035  09590   TOPCON==<ATNTK-ONEFUN>*2+1
 5201  006263' 001000  000376    09600           CPI     TOPCON          ;BIGGER THAN ARC-TANGENT?
 5202  006264' 000000  000035
 5203  006265' 001000  000345    09620           PUSH    H               ;SAVE THE FUNCTION NUMBER
 5204  006266' 0F1000  000334    09640           CC      FNCSNG          ;IF NOT, FORCE FAC TO SINGLE-PRECISION
 5205  006267' 000000  0000C4*
 5206  006270' 000000  006C61'
 5207  006271' 001000  000341    09660           POP     H               ;RESTORE THE FUNCTION NUMBER
 5208  006272' 0F0000  09680   NOTFRF: >
 5209  006273' 001000  000021    09700           LXI     D,LABBCK        ;RETURN ADDRESS
 5210  006274' 000000  006162'
 5211  006275' 000000  006167'
```

```
                                                    ;MAKE THEM REALLY COME BACK
                        PUSH    D
                FINGO:  LXI     B,FUNUSR            ;FUNCTION DISPATCH TABLE
                        DAD     B                   ;ADJ ON THE OFFSET
                IFE     LENGTH,<
                        PUSHM
                        RET>                        ;GO TO THE ADDRESS POINTED TO BY (M,L)
                IFN     LENGTH,<
                        MOV     C,M                 ;FASTER THAN PUSHM
                        INX     H
                        MOV     H,M
                        MOV     L,C
                        PCHL>                       ;GO PERFORM THE FUNCTION
                IFE     LENGTH-2,<
                ;
                ; GET THE VALTYP AND SET CONDITION CODES AS FOLLOWS:
                ; CONDITION CODE        TRUE SET        FALSE SET
                ;
                ; SIGN                  INT=2           STR,SNG,DBL
                ; ZERO                  STR=3           INT,SNG,DBL
                ; ODD PARITY            SNG=4           INT,STR,DBL
                ; NO CARRY              DBL=10          INT,STR,SNG
                ;
GETYPE: LDA     VALTYP                              ;GET THE VALTYP
                CPI     8                           ;SET CARRY CORRECTLY
                DCR     A                           ;SET THE OTHER CONDITION CODES CORRECTLY
                DCR     A                           ; WITHOUT AFFECTING CARRY
                DCR     A
                RET>
                IFN     LENGTH-2,<
                IFN     LENGTH,<
ORI:    XWD     "U1000,"O366                        ;"ORI" , FLAG AS "OR"
                                                    ;AND LSE COMMON "AND" CODE
ANDI:   XRA     A                                   ;FLAG AS "AND"
ANDCONT:
                PUSH    PSW
                IFN     STRING,<CALL    CHKNUM>
                CALL    DEINT                       ;GET (D,E)=INT VALUE AND CHECK SIZE
                POP     PSW
                XCHG                                ;(H,L)=INT VALUE
                POP     B                           ;GET HIGH ORDER OFF
                XTHL                                ;PUT INT VALUE ON
                                                    ;GET LOW OF SECOND ARG OFF
                XCHG                                ;(D,E)=LOW OF SECOND ARG
                CALL    MKTFR
                PUSH    PSW
                CALL    DEINT                       ;GET (D,E)=INT VALUE
                POP     PSW
```



```
                                                    ;OF FIRST ARG AND CHECK SIZE
                        POP     B                   ;(B,C)=INT VALUE OF SECOND ARG
                        MOV     A,C                 ;(A)=LOW OF SECOND ARG
                        LXI     H,GIVACF            ;SETUP JUMP ADDRESS
                        JNZ     ORFIN               ;IT WAS "OR" SO FINISH UP
                        ANA     E                   ;AND TWO LOW ORDERS
                        MOV     C,A                 ;SAVE ANSWER IN (C)
                        MOV     A,B                 ;(A)=HIGH ORDER SECOND ARG
                        ANA     D                   ;(A)=HIGH ORDER OF ANSWER
                        PCHL                        ;FLOAT (A,C) AS ANSWER
ORFIN:  ORA     E                                   ;OR TWO LOW ORDERS
                        MOV     C,A                 ;SAVE ANSWER LOW ORDER IN (C)
                        MOV     A,B                 ;(A)=HIGH ORDER SECOND ARG
                        ORA     D                   ;OR TWO HIGH ORDERS
                                                    ;(A)=HIGH ORDER OF ANSWER
                        PCHL                        ;FLOAT (A,C) AS ANSWER
FINREL: LXI     H,PTDORL                            ;MAKE (H,L) POINT AT OPERATOR ADDRESS
                IFN     STRING,<
                        LDA     VALTYP              ;RTURE VALUE TYPE AS LOW
                        RAR                         ;ORDER BIT OF (E)
                        MOV     A,D                 ;GET RELATIONAL MEMORIES IN (A)
                        RAL                         ;MOVE CARRY BIT IN
                        MOV     E,A>                ;KEEP THIS BYTE IN (E)
                IFE     STRING,<MOV     E,D>
                        MVI     D,100               ;PRECEDENCE OF ALL RELATIONAL
                                                    ;OPERATORS IS 100
                        MOV     A,B                 ;(A)=PRECEDENCE OF OLD OPERATOR
                        CMP     D                   ;SEE WHETHER TO APPLY OLD OPERATOR
                        RNC                         ;IF OLD OPERATOR HAS EQUAL OR GREATER
                                                    ;PRECEDENCE THAN IT MUST BE APPLIED
                        JMP     DOPREC              ;SEE IF TIME TO APPLY
                                                    ;AND IF NOT SAVE INFO ON THE STACK
                ;
PTDORL: ADR(DOREL)                                  ;ADDRESS OF RELATIONAL
                                                    ;OPERATOR APPLILATION
                                                    ;ROUTINE
                ;
                ; TIME TO PERFORM A RELATIONAL OPERATOR
                ; (C) CONTAINS THE BITS AS TO WHICH RELATIONAL
                ; OPERATOR IT WAS (IF STRINGS ON
                ; LOW ORDER BIT SAYS WHETHER IT WAS STRING OR NOT)
                ;
DOREL:  MOV     A,C                                 ;GET MEMORIES INTO (A)
                IFN     STRING,<
                        ORA     A
                        RAR>                        ;CARRY=WHETHER WAS STRING OR NOT
                        POPR                        ;POP OFF LEFT RESULT
                        PUSH    PSW                 ;SAVE WHICH OPERATOR IT WAS
                IFE     STRING,<
                        CALL    FCOMP>              ;DO A NUMERIC COMPARE
                IFN     STRING,<
                        CALL    CHKVAL              ;SEE IF VALTYP MATCHES
                                                    ;CARRY AND SET ZERO IN THE
```

```
                                                11720                           ;NUMERIC CASE
                                                11740           LXI     H,DOCMP         ;COME BACK TO COMPARE
                                                11760           PUSH    H
                                                11780           JZ      FCOMP           ;DO NUMERIC COMPARE
                                                11800           XRA     A               ;SET VALUE TYPE AS NUMERIC
                                                11820           STA     VALTYP>>>
                                                11840   IFN     STRING,<
                                                11860   ;
                                                11880   ; THE FOLLOWING ROUTINE COMPARES TWO STRINGS
                                                11900   ; ONE WITH DESC IN [D,E] OTHER WITH DESC. IN [FACLO, FACLO+1]
                                                11920   ; A=0 IF STRINGS EQUAL
                                                11940   ; A=377 IF B,C,D,E POINTER FACLO
                                                11960   ; A=1 IF B,C,D,E .LT. FACLO
                                                11980   ;
                                                12000   IFN     LENGTH=2,<
                                                12020   STRCMP: PUSH    D               ;SAVE DESC. POINTER TO FIRST STR.
                                                12040           CALL    FREFAC          ;FREE THE FACLO STR
                                                12060           POP     D               ;RESTORE 1ST DESC. POINTER
                                                12080           PUSHM                   ;SAVE LENGTH
                                                12100           PUSHM                   ;SAVE POINTER
                                                12120           CALL    FRETMP          ;FREES 1ST DESC. POINTER
                                                12140           CALL    MOVRM           ;[B,C] POINT AT FIRST CHAR
                                                12160                                   ;[E] HAS THE LENGTH
                                                12180           POP     H               ;GET 2ND CHARACTER POINTER IN H
                                                12200           XTHL                    ;GET 2ND CHARACTER COUNTER IN L
                                                12220           MOV     D,L>            ;SAVE IN D
                                                12240   IFE     LENGTH-2,<
0006326' 001000  000315         12260   STRCMP: CALL    FRESTR          ;FREE UP THE FAC STRING, AND GET THE
0006321' 000000  010431'
0006322' 000000  006310'
                                                12280                                   ;POINTER TO THE FAC DESCRIPTOR IN [H,L]
0006325' 001000  000176         12300           MOV     A,M             ;SAVE THE LENGTH OF THE FAC STRING IN [A]
0006324' 001000  000043         12320           INX     H
0006325' 001000  000116         12340           MOV     C,M             ;SAVE THE POINTER AT THE FAC STRING
                                12342                                   ;DATA IN [B,C]
0006326' 001000  000043         12360           INX     H
0006327' 001000  000106         12380           MOV     B,M
0006330' 001000  000321         12400           POP     D               ;GET THE STACK STRING POINTER
0006331' 001000  000305         12420           PUSH    B               ;SAVE THE POINTER AT THE FAC STRING DATA
0006332' 001000  000365         12440           PUSH    PSW             ;SAVE THE FAC STRING LENGTH
0006333' 001000  000315         12460           CALL    FRETMP          ;FREE UP THE STACK STRING AND RETURN
0006334' 000000  010440'
0006335' 000000  006321'
                                12480                                   ;THE POINTER TO THE STACK STRING DESCRIPTOR
                                12500                                   ;IN [H,L]
0006336' 001000  000321         12520           POP     D               ;[D]=LENGTH OF FAC STRING
0006337' 001000  000136         12540           MOV     E,M             ;[E]=LENGTH OF STACK STRING
0006340' 001000  000043         12560           INX     H
0006341' 001000  000116         12580           MOV     C,M             ;[B,C]=POINTER AT STACK STRING
0006342' 001000  000043         12600           INX     H
0006343' 001000  000106         12620           MOV     B,M>
0006344' 001000  000341         12640           POP     H               ;GET BACK 2ND CHARACTER POINTER
0006345' 001000  000173         12660   CSLOOP: MOV     A,E             ;BOTH STRINGS ENDED
```



```
0006346' 001000  000262         12680           ORA     D               ;TEST BY OR'ING THE LENGTHS TOGETHER
0006347' 001000  000310         12700           RZ                      ;IF SO, RETURN WITH A ZERO
0006350' 001000  000172         12720           MOV     A,D             ;GET FACLO STRING LENGTH
0006351' 001000  000267         12740           ORA     A               ;IF IT ENDED, OTHER MUST NOT HAVE
0006352' 001000  000057         12760           CMA                     ;MAKE -1
0006353' 001000  000310         12780           RZ                      ;TEST
0006354' 001000  000257         12800           XRA     A               ;MUST NOT HAVE BEEN ZERO, TEST CASE
0006355' 001000  000273         12820           CMP     E               ;OF B,C,D,E STRING HAVING ENDED FIRST
0006356' 001000  000074         12840           INR     A               ;RETURN WITH A=1
0006357' 001000  000320         12860           RNC                     ;TEST THE CONDITION
                                12880   ;HERE WHEN NEITHER STRING ENDED
0006360' 001000  000025         12900           DCR     D               ;DECREMENT BOTH CHARACTER COUNTS
0006361' 001000  000035         12920           DCR     E
0006362' 001000  000012         12940           LDAX    B               ;GET CHARACTER FROM B,C,D,E STRING
0006363' 001000  000276         12960           CMP     M               ;COMPARE WITH FACLO STRING
0006364' 001000  000043         12980           INX     H               ;BUMP POINTERS (INX DOESNT CLOBBER CC'S)
0006365' 001000  000003         13000           INX     B
0006366' 001000  000312         13020           JZ      CSLOOP          ;IF BOTH THE SAME, MUST BE MORE TO STRINGS
0006367' 000000  006345'
0006370' 000000  006334'
0006371' 001000  000077         13040           CMC                     ;HERE WHEN STRINGS DIFFER
0006372' 001000  000303         13060           JMP     SIGNS>          ;SET [A] ACCORDING TO CARRY
0006373' 000000  000004*
0006374' 000000  006367'
                                13080   IFN     LENGTH,<
0006375' 001000  000074         13100   DOCMP:  INR     A               ;SETUP BITS
0006376' 001000  000017         13120           RRC     A               ;A=LESS 2=EQUAL 1=GREATER
0006377' 001000  000301         13140           POP     B               ;WHAT DID HE WANT?
0006400' 001000  000240         13160           ANA     B               ;ANY BITS MATCH?
0006401' 001000  000306         13180           ADI     255             ;MAP 0 TO 0
0006402' 000000  000377
0006403' 001000  000257         13200           SBB     A               ;AND ALL OTHERS TO 377
                                13220   IFE     LENGTH-2,<
0006404' 001000  000315         13240           CALL    CONIA##         ;CONVERT [A] TO AN INTEGER SIGNED
0006405' 000000  000004*
0006406' 000000  006373'
0006407' 001000  000303         13260           JMP     RETAOP>         ;RETURN FROM OPERATOR APPLICATION
0006410' 000000  005354'
0006411' 000000  006405'
                                13280   IFN     LENGTH-2,<
                                13300           JMP     FLOAT           ;MAKE FAC=[A] SIGNED
                                13320                                   ;COULD FALL INTO FLOAT BUT MESSY (SAVES
                                13340                                   ;TWO BYTES)
                                13360
                                13380   NOTER1  MVI     D,90            ;NOTE HAS PRECEDENCE 90
                                13400           CALL    LPOPER          ;GO PERFORM
                                13420   IFN     STRING,<CALL    CHKNUM> ;MAKE SURE ITS INT
                                13440           CALL    DEINT           ;GET VALUE IN [D,E]
                                13460           MOV     A,E
                                13480           CMA                     ;COMPLEMENT
                                13500           MOV     C,A             ;[C] LOW ORDER OF ANSWER
                                13520           MOV     A,D
                                13540           CMA                     ;COMPLEMENT HIGH ORDER TOO
```

```
                                        13580           CALL    GIVACF          ;FLOAT [A,C] AS RESULT
                                        13590           POP     B               ;TAKE RETURN ADDRESS OF FRMEVL
                                        13600           JMP     RETADP>>        ;OFF AND RETURN TO THE RIGHT
                                        13620                                   ;PLACE SO THE TEXT POINTER
                                        13630                                   ;WILL GET SET UP TO WHAT IT WAS
                                        13640                                   ;WHEN LPOPER RETURNED.
                                        13660           IFE     LENGTH-2,<
0064162' 001000  000026                 13700   NOTER1: MVI     D,90            ;NOT' HAS PRECEDENCE 90, SO
0064163' 000000  000132
0064164' 001000  000315                 13720           CALL    LPOPER          ;FORMULA EVALUATION IS ENTERED WITH A DUMMY
0064165' 000000  005341'
0064166' 000000  006410'
                                        13730                                   ;ENTRY OF 90 ON THE STACK
0064167' 001000  000315                 13740           CALL    FRCINT          ;COERCE THE ARGUMENT TO INTEGER
0064170' 000000  005573'
0064171' 000000  006415'
0064172' 001000  000175                 13760           MOV     A,L             ;COMPLEMENT [H,L]
0064173' 001000  000057                 13780           CMA
0064174' 001000  000157                 13800           MOV     L,A
0064175' 001000  000174                 13820           MOV     A,H
0064176' 001000  000057                 13840           CMA
0064177' 001000  000147                 13860           MOV     H,A
0064200' 001000  000042                 13880           SHLD    FACLO           ;UPDATE THE FAC
0064201' 000000  001637'
0064202' 000000  006420'
0064203' 001000  000301                 13900           POP     B               ;FRMEVL, AFTER SEEING THE PRECEDENCE
                                        13910                                   ;OF 90 THINKS IT IS APPLYING AN OPERATOR
                                        13915                                   ;SO IT HAS THE TEXT POINTER IN TEMP2 SO
                                        13920           JMP     RETADP          ;RETURN TO REFETCH IT
0064204' 001000  000303                 13940           JMP     RETADP          ;RETURN TO REFETCH IT
0064205' 000000  005354'
0064206' 000000  006431'
                                        14042   ;
                                        14044   ; DANDOR APPLIES THE "AND" AND "OR" OPERATORS
                                        14046   ; AND SHOULD BE USED TO IMPLEMENT ALL LOGICAL OPERATORS,
                                        14048   ; WHENEVER AN OPERATOR IS APPLIED, ITS PRECEDENCE IS IN [B],
                                        14050   ; THIS FACT IS USED TO DISTINGUISH BETWEEN "AND" AND "OR".
                                        14052   ; THE RIGHT HAND ARGUMENT IS COERCED TO INTEGER, JUST AS
                                        14054   ; THE LEFT HAND ONE WAS WHEN IT WAS PUSHED ON THE STACK.
                                        14056   ;
0064207' 001000  000305                 14060   DANDOR: PUSH    B               ;SAVE THE PRECEDENCE "OK"=70
0064210' 001000  000315                 14080           CALL    FRCINT          ;COERCE RIGHT HAND ARGUMENT TO INTEGER
0064211' 000000  006420*
0064212' 000000  006433'
0064213' 001000  000361                 14100           POP     PSW             ;GET BACK THE PRECEDENCE TO DISTINGUISH
                                        14120                                   ;"AND" AND "OR"
0064214' 001000  000321                 14140           POP     D               ;POP OFF THE LEFT HAND ARGUMENT
0064215' 001000  000376                 14160           CPI     70              ;SET ZERO FOR "OR"
0064216' 000000  000106
0064217' 001000  000173                 14180           MOV     A,E             ;SETUP LOW IN [A]
0064220' 001000  000312                 14200           JZ      ORFIN           ;DO "OR" IF PRECEDENCE WAS 70
0064221' 000000  006463'
0064222' 000000  006441'
```



```
0064223' 001000  000245                 14220           ANA     L
0064224' 001000  000157                 14240           MOV     L,A
0064225' 001000  000174                 14260           MOV     A,H
0064226' 001000  000242                 14280           ANA     D
0064227' 001000  000147                 14300           MOV     H,A
0064230' 001000  000303                 14320           JMP     MAKINT          ;RETURN THE INTEGER [H,L]
0064231' 000000  000000*
0064232' 000000  006451'
                                        14322                                   ; AS THE "AND"ED RESULT
0064233' 001000  000265                 14324   ORFIN:  ORA     L
0064234' 001000  000157                 14326           MOV     L,A
0064235' 001000  000174                 14328           MOV     A,H
0064236' 001000  000262                 14330           ORA     D
0064237' 001000  000147                 14332           MOV     H,A
0064240' 001000  000303                 14334           JMP     MAKINT>         ;RETURN THE INTEGER [H,L]
0064241' 000000  006461*
0064242' 000000  006461'
                                        14336                                   ; AS THE "OR"ED RESULT
                                        14340   PAGE
```

```
                                14360   SUBTTL  DIMENSION & VARIABLE SEARCHING
                                14380   DIMCON: DCX     H               ;SEE IF COMMA ENDED THIS VARIABLE
                                14400           CHRGET
                                14420           RZ                      ;IF TERMINATOR, GOOD BYE
                                14440           SYNCHK  44              ;MUST BE COMMA
                                14442   ;
                                14444   ; THE "DIM" CODE SETS DIMFLG AND THEN FALLS INTO THE VARIABLE
                                14446   ; SEARCH ROUTINE. THE VARIABLE SEARCH ROUTINE LOOKS AT
                                14448   ; DIMFLG AT THREE DIFFERENT POINTS:
                                14450   ;
                                14452   ;       1) IF AN ENTRY IS FOUND, DIMFLG BEING ON INDICATES
                                14454   ;               A "DOUBLY DIMENSIONED" VARIABLE
                                14456   ;       2) WHEN A NEW ENTRY IS BEING BUILT DIMFLG'S BEING ON
                                14458   ;               INDICATES THE INDICES SHOULD BE USED FOR
                                14460   ;               THE SIZE OF EACH INDICE. OTHERWISE THE DEFAULT
                                14462   ;               OF TEN IS USED.
                                14464   ;       3) WHEN THE BUILD ENTRY CODE FINISHES, ONLY IF DIMFLG IS
                                14466   ;               OFF WILL INDEXING BE DONE
                                14468   ;
0005500' 021000  000001         14470   DIM:    LXI     B,DIMCON        ;PLACE TO COME BACK TO
0005501' 000000  006473'
0005502' 000000  006471'
                                14480           PUSH    B
0005504' 021000  000366         14500           XWD     "O1000,"O366    ;"DIM" NON ZERO THING
                                14520   ;
                                14540   ; ROUTINE TO READ THE VARIABLE NAME AT THE CURRENT TEXT POSITION
                                14560   ; AND PUT A POINTER TO ITS VALUE IN [D,E]. [H,L] IS UPDATED
                                14580   ; TO POINT TO THE CHARACTER AFTER THE VARIABLE NAME.
                                14600   ; VALTYP IS SETUP. NOTE THAT EVALUATING SUBSCRIPTS IN
                                14620   ; A VARIABLE NAME CAN CAUSE RECURSIVE CALLS TO PTRGET SO AT
                                14640   ; THAT POINT ALL VALUES MUST BE STORED ON THE STACK.
                                14660   ;
0005507' 021000  002257         14680   PTRGET: XRA     A               ;MAKE [A]=0
0005510' 000000  000052         14700           STA     DIMFLG          ;FLAG IT AS SUCH
0005511' 000000  001542'
0005513' 000000  000201'
0005515' 021000  000120         14720           MOV     B,M             ;GET FIRST CHARACTER IN [B]
0005516' 021000  000315         14740   PTRGT2: CALL    ISLET           ;CHECK FOR LETTER
0005519' 000000  003612'
0005521' 000000  006527'
0005523' 021000  000332         14760           JC      $NERR           ;MUST HAVE A LETTER
0005526' 021000  000257         14780           XRA     A
0005527' 001000  000117         14800           MOV     C,A             ;ASSUME NO SECOND CHARACTER
                                14820   IFN     LENGTH-2,<
                                14840   IFN     STRING,<
                                14860           STA     VALTYP>>        ;DEFAULT IS ZERO (NUMERIC)
0005532' 021000  000327         14880           CHRGET                  ;GET THE FOLLOWING CHARACTER
                                14900   IFE     LENGTH,<
                                14920           JNC     NOSEC>          ;ONLY NUMBERS ALLOWED
                                14940   IFN     LENGTH,<
```



```
0005535' 021000  000332         14960           JC      ISSEC           ;CARRY SET BY CHRGET IF CHARACTER IS
0005536' 000000  006534'
0005537' 000000  006516'
                                14980                                   ;NUMERIC
0005540' 021000  000315         15000           CALL    ISLET           ;SET CARRY IF NOT ALPHABETIC
0005543' 000000  003612'
0005545' 000000  006524'
0005547' 021000  000332         15020           JC      NOSEC>          ;ALLOW ALPHABETICS
0005552' 000000  006547'
0005554' 000000  006527'
0005556' 021000  000117         15040   ISSEC:  MOV     C,A             ;IT IS A NUMBER--SAVE IN C
0005557' 021000  000327         15060   EATEM:  CHRGET                  ;LOOK AT NEXT CHARACTER
                                15080   IFN LENGTH,<
0005562' 021000  000332         15100           JC      EATEM           ;SKIP NUMERICS
0005565' 000000  006535'
0005567' 000000  006536'
0005571' 021000  000315         15120           CALL    ISLET
0005574' 000000  003612'
0005576' 000000  006537'
0005600' 021000  000322         15140           JNC     EATEM>          ;SKIP ALPHABETICS
0005603' 000000  006535'
0005605' 000000  006542'
0005607'                        15160   NOSEC:
                                15180   IFE     LENGTH-2,<
0005607' 021000  000021         15200           LXI     D,MAVTYP        ;SAVE JUMPS BY USING RETURN ADDRESS
0005610' 000000  006574'
0005612' 000000  006545'
0005614' 021000  000325         15220           PUSH    D
0005615' 001000  000026         15240           MVI     D,8             ;ASSUME IT'S DOUBLE PRECISION
0005616' 000000  000010
0005620' 001000  000376         15260           CPI     "#"             ;CHECK THE CHARACTER
0005622' 000000  000043
0005624' 001000  000310         15280           RZ                      ;WHEN WE MATCH, SETUP VALTYP
0005625' 001000  000026         15300           MVI     D,2             ;CHECK FOR INTEGER
0005627' 000000  000002
0005631' 001000  000376         15320           CPI     "%"
0005633' 000000  000045
0005635' 001000  000310         15340           RZ
0005636' 001000  000024         15360           INR     D               ;CHECK FOR STRING
0005637' 001000  000376         15380           CPI     "$"
0005641' 000000  000044
0005643' 001000  000310         15400           RZ
0005644' 001000  000024         15420           INR     D               ;SINGLE PRECISION IS THE DEFAULT
0005645' 001000  000053         15440           DCX     H               ;AND MARKING CHARACTER
0005646' 001000  000311         15460           RET                     ;GET RID OF RETURN ADDRESS
0005647' 001000  000172         15480   MAVTYP: MOV     A,D             ;SETUP VALTYP
0005650' 001000  000006         15500           STA     VALTYP
0005652' 000000  001543'
0005654' 000000  006550'
0005656' 021000  000327         15520           CHRGET>                 ;READ PAST TYPE MARKER
                                15540   IFN     LENGTH-2,<
                                15560   IFN     STRING,<
                                15580           SUI     "$"             ;IS IT A STRING?
```

```
                                15600           JNZ     NOTSTR          ;IF NOT VALTYP ALREADY=0
                                15620           INR     A               ;[A]=1
                                15640           STA     VALTYP          ;FLAG THIS AS A STRING
                                15660           RRC                     ;MAKE [A]=128
                                15680           ADD     C               ;MAKE [A]=SECOND CHARACTER
                                15700           MOV     C,A             ;BACK INTO [C] WITH STRING BIT ON
                                15720           CHRGET                  ;GET CHARACTER AFTER "$"
                                15740   NOTSTR: >>
                                15760           IFN     LENGTH,<
                                15780           LDA     SUBFLG          ;GET FLAG WHETHER TO ALLOW ARRAYS
                                15800           ADD     M>              ;ADD ONTO CHARACTER
                                15820           CPI     "("             ;ARRAY PERHAPS (IF SUBFLG SET NEVER WILL MATCH)
                                15840           JZ      ISARY           ;IT IS!
                                15860   IFN     LENGTH,<
                                15880           XRA     A               ;ALLOW PARENS AGAIN
                                15900           STA     SUBFLG>         ;SAVE IN FLAG LOCATION
                                15920           PUSH    H               ;SAVE THE TEXT POINTER
                                15940   IFE     LENGTH-2,<
                                15980           MOV     A,D             ;VALUE TYPE INTO [A]
                                15990           LHLD    VARTAB          ;[H,L]=PLACE TO START THE SEARCH
                                16000   LOPFND: PUSH    PSW             ;SAVE THE VALUE TYPE
                                16020           XCHG                    ;[D,E]=POINTER INTO SIMPLE VARIABLES
                                16040           LHLD    ARYTAB          ;[H,L]=END OF SIMPLE VARIABLES
                                16060           COMPAR                  ;SEE IF THE END HAS BEEN REACHED
                                16080           POP     H               ;[H]=VALTYP
                                16100           JZ      NOTFNS          ;COULDN'T FIND IT, SO MAKE A NEW ENTRY
                                16120           LDAX    D               ;GET THE VALTYP OF THIS SIMPLE VARIABLE
                                16140           MOV     L,A             ;SAVE SO WE KNOW HOW MUCH TO SKIP
                                16160           CMP     M               ;COMPARE WITH OUR VALTYP
                                16180           INX     D
                                16200           JNZ     NOTIT1          ;NOT RIGHT KIND == SKIP IT
                                16220           LDAX    D               ;[A]=FIRST CHARACTER OF THIS VARIABLE
                                16240           CMP     C               ;SEE IF OUR VARIABLE MATCHES
                                16260           JNZ     NOTIT1
```



```
                                16300           LDAX    D               ;SEE IF SECOND LHARACTER MATCHES
                                16320           CMP     B
                                16340           JZ      FINPTR          ;THAT WAS IT, ALL DONE
                                16360           XWD     "01000,"076     ;"MVI A," AROUND THIS INX SINCE THE POINTER
                                16380                                   ;IS ALREADY INCREMENTED
                                16400   NOTIT1: INX     D
                                16420           INX     D
                                16440           MOV     A,M             ;SKIP OVER THE
                                16460                                   ;CURRENT VARIABLE SINCE WE DIDN'T MATCH
                                16480                                   ;SAVE THE VALTYP IN [A]
                                16500           MVI     M,0             ;[M,L]=NUMBER OF BYTES TO SKIP
                                16520           DAD     D               ;ADD ON THE POINTER
                                16540           JMP     LOPFND          ;AND SEARCH SOME MORE
                                16560   NOTFNS: PUSH    B               ;SAVE THE LOOKS
                                16580           MOV     C,M             ;[B,C]=LENGTH OF THIS VARIABLE
                                16600           MOV     B,A             ;[B]=0
                                16620           PUSH    B               ;SAVE THE VALTYP ON THE STACK
                                16640           INX     B               ;MAKE THE LENGTH INCLUDE
                                16660                                   ;THE LOOKS TOO
                                16680           INX     B
                                16700           INX     B>
                                16720   IFN     LENGTH-2,<
                                16740           LHLD    ARYTAB          ;PLACE TO STOP SEARCHING
                                16760           XCHG
                                16780           LHLD    VARTAB          ;GET THE PLACE TO START
                                16800   LOPFND: COMPAR                  ;SEE IF WE ARE THERE
                                16820           JZ      NOTFNS          ;COULDN'T FIND THIS VARIABLE
                                16840                                   ;SO MAKE ROOM FOR IT
                                16860           MOV     A,C
                                16880           SUB     M               ;IS THIS VARIABLE THE ONE?
                                16900           INX     H
                                16920           JNZ     NOTIT           ;NOPE
                                16940           MOV     A,B
                                16960           SUB     M               ;TRY SECOND CHARACTER MATCHING
                                16980   NOTIT1: INX     H
                                17000           JZ      FINPTR          ;THAT WAS IT!
                                17020           INX     H               ;SKIP OVER THAT ONE--NOT IT
                                17040           INX     H
                                17060           INX     H
                                17080           INX     H
                                17080           JMP     LOPFND          ;TRY AGAIN
                                17100   NOTFNS: PUSH    B               ;REMEMBER WHAT THIS
                                17120                                   ;VARIABLE LOOKS LIKE
                                17140           LXI     B,6+$CODE>      ;THE AMOUNT TO SMOVE
                                17160                                   ;EVERYTHING UP BY
                                17180           LHLD    STREND          ;THE CURRENT END OF STORAGE
```

```
5708  006705' 001000  000345          17200          PUSH    H               ;SAVE THIS #
5709  006706' 001000  000011          17220          DAD     B               ;ADD ON THE AMOUNT OF SPACE
5710                                  17240                                  ;EXTRA NOW BEING USED
5711  006707' 001000  000301          17260          POP     B               ;POP OFF HIGH ADDRESS TO MOVE
5712  006708' 001000  000345          17280          PUSH    H               ;SAVE NEW CANDIDATE FOR STREND
5713  006709' 001000  000315          17300          CALL    BLTU            ;BLOCK TRANSFER AND MAKE SURE
5714  006710' 000000  002035'
5715  006711' 000000  006741'
5716                                  17320                                  ;WE ARE NOT OVERFLOWING THE
5717                                  17340                                  ;STACK SPACE
5718  006712' 001000  000341          17360          POP     H               ;[H,L]=NEW STREND
5719  006713' 001000  000042          17380          SHLD    STREND          ;STORE SINCE WAS OK
5720  006714' 000000  001625'
5721  006715' 000000  006710'
5722                                  17400                                  ;THERE WAS ROOM, AND BLOCK TRANSFER
5723                                  17420                                  ;WAS DONE, SO UPDATE POINTERS
5724  006716' 001000  000140          17440          MOV     H,B             ;GET BACK [H,L] POINTING AT THE END
5725  006717' 001000  000151          17460          MOV     L,C             ;OF THE NEW VARIABLE
5726  006720' 001000  000042          17480          SHLD    ARYTAB          ;UPDATE THE ARRAY TABLE POINTER
5727  006721' 000000  001625'
5728  006722' 000000  006714'
5729  006723' 001000  000053          17500  ZEROEK: DCX     H               ;[H,L] IS RETURNED POINTING TO THE
5730  006724' 001000  000066          17520          MVI     M,0             ;END OF THE VARIABLE SO WE
5731  006725' 000000  000000
5732  006726' 001000  000347          17540          COMPAR                  ;ZERO BACKWARDS TO [D,E] WHICH
5733  006727' 001000  000320          17560          JNZ     ZEROEK          ;POINTS TO THE START OF THE VARIABLE
5734  006730' 000000  006723'
5735  006731' 000000  006721'
5736                                  17580  IFE     LENGTH-2,<
5737  006732' 001000  000321          17600          POP     D               ;[E]=VALTYP
5738  006733' 001000  000163          17620          MOV     M,E             ;STORE AS PART OF THE LOOKS
5739  006734' 001000  000043          17640          INX     H>
5740  006735' 001000  000321          17660          POP     D
5741  006736' 001000  000163          17680          MOV     M,E             ;PUT DESCRIPTION
5742  006737' 001000  000043          17700          INX     H
5743  006740' 001000  000162          17720          MOV     M,D             ;OF THIS VARIABLE
5744                                  17740                                  ;INTO MEMORY
5745                                  17760  IFE     LENGTH-2,<
5746  006741' 001000  000353          17780          XCHG                    ;POINTER AT VARIABLE INTO [D,E]
5747  006742' 001000  000023          17800  FINPTR: INX     D>              ;POINT AT THE VALUE
5748                                  17820  IFN     LENGTH-2,<
5749                                  17840          INX     H
5750                                  17860  FINPTR: XCHG>                   ;VARIABLE POINTER INTO [D,E]
5751  006743' 001000  000341          17880          POP     H               ;RESTORE THE TEXT POINTER
5752  006744' 001000  000311          17900          RET
5753
5754                                  17940  IFE     MULDIM,<
5755                                  17960  ISARY:  PUSH    B               ;REMEMBER WHAT VARIABLE LOOKS
5756                                  17980                                  ;LIKE
5757                                  18000  IFN     STRING,<
5758                                  18020          PUSH    H               ;SAVE THE TXTPTR
5759                                  18040          LHLD    DIMFLG          ;[L]=DIMFLG [H]=VALTYP
5760                                  18060          XTHL>                   ;PUT VALTYP AND DIMFLG ON THE STACK
```



```
5761                                  18080                                  ;AND RESTORE THE TEXT POINTER
5762                                  18100  IFE     STRING,<
5763                                  18120          LDA     DIMFLG          ;SINCE THIS CALL IS RECURSIVE
5764                                  18140          PUSH    PSW>            ;DIMFLG MUST BE SAVED ON THE STACK
5765                                  18160          CALL    INTIDX          ;EVALUATE THE INDEX INTO [D,E]
5766                                  18180          SYNCHK  ")"             ;MAKE SURE WE CLOSED IT
5767                                  18200  IFN     STRING,<
5768                                  18220          XTHL                    ;[L]=DIMFLG [H]=VALTYP
5769                                  18240                                  ;TEXT POINTER ONTO THE STACK
5770                                  18260          SHLD    DIMFLG          ;SAVE BOTH VALUES BACK
5771                                  18280          POP     H>              ;RESTORE THE TEXT POINTER
5772                                  18300  IFE     STRING,<
5773                                  18320          POP     PSW             ;GET DIMFLG OFF THE STACK
5774                                  18340          STA     DIMFLG>         ;RESTORE IT
5775                                  18360          XTHL                    ;[H,L] GET VARIABLE DESCRIPTOR
5776                                  18380                                  ;TEXT POINTER IS PUT ONTO
5777                                  18400                                  ;THE STACK
5778                                  18420          XCHG                    ;[D,E]=DESCRIPTOR
5779                                  18440                                  ;[H,L]=INDEX
5780                                  18460          DAD     H               ;MULTIPLY BY 4 TO GET
5781                                  18480          DAD     H               ;BYTE OFFSET
5782                                  18500          PUSH    H               ;SAVE THE INDEX
5783                                  18520          LHLD    ARYTAB          ;PLACE TO START SEARCH
5784                                  18540          XWD     "D1200,1        ;LXI B,* OVER THE NEXT 2
5785                                  18560  LOPFD2: POP     B               ;[B,C]=LENGTH OF LAST VARIABLE
5786                                  18580          DAD     B               ;SKIP OVER LAST VARIABLE BY ADDING ITS
5787                                  18600                                  ;LENGTH ONTO [H,L]
5788                                  18620          XCHG                    ;[D,E] GET CURRENT SEARCH POINT
5789                                  18640          PUSH    H               ;SAVE THE VARIABLE LOOK
5790                                  18660          LHLD    STREND          ;GET PLACE TO STOP
5791                                  18680          COMPAR                  ;SEE IF WE ARE THERE
5792                                  18700          XCHG                    ;[H,L] GETS SEARCH POINT
5793                                  18720          POP     D               ;POP OFF VARIABLE LOOKS
5794                                  18740          JZ      NOTFDD          ;COULDN'T FIND IT
5795                                  18760          PUSHM                   ;PUT ON LOOKS OF VARIABLE
5796                                  18780                                  ;WE ARE EXAMINING
5797                                  18800          XTHL                    ;PUT [H,L] ON THE STACK AND
5798                                  18820                                  ;LOOKS OF VARIABLE WE ARE
5799                                  18840                                  ;EXAMINING INTO [H,L]
5800                                  18860          COMPAR                  ;IS THIS THE VARIABLE
5801                                  18880          POP     H               ;POP OFF SEARCH POINTER
5802                                  18900          PUSHM                   ;PUSH LENGTH OF VARIABLE
5803                                  18920                                  ;BEING EXAMINED ONTO THE STACK
5804                                  18940          JNZ     LOPFD2          ;IF NO MATCH,GO LOOK SOMEMORE
5805                                  18960          LDA     DIMFLG          ;IS THIS VARIABLE TRYING TO BE
5806                                  18980                                  ;DIMENSIONED AND ALREADY
5807                                  19000          ORA     A               ;EXISTS?
5808                                  19020          MVI     E,ERRDD         ;THATS ERROR ERRD,
5809                                  19040          JNZ     ERROR
5810                                  19060  MAKOFN: POP     D               ;POP OFF LENGTH OF THIS VARIABLE
5811                                  19080          DCX     D               ;DECREMENT LENGTH SO WE CAN
5812                                  19100                                  ;JUST LOOK AT "CARRY" AFTER
5813                                  19120                                  ;CALLING COMPAR
```

```
                                19140           XTHL                    ;TRADE POINTER AT VARIABLE WITH
                                19150                                   ;INDEX INTO THE VARIABLE
                                19160           COMPAR                  ;SEE IF INDEX IS TOO BIG
                                19200           MVI     E,ERRBS         ;THATS ERROR ERRBS
                                19220           JNC     ERROR           ;SINCE LENGTH REALLY HAS AN
                                19240                                   ;EXTRA ONE ADDED TO IT
                                19260                                   ;IF INDEX=LENGTH DOESN'T CARRY
                                19280                                   ;HE IS IN TROUBLE
                                19300           POP     D               ;POP OFF POINTER AT VARIABLE
                                19320           DAD     D               ;ADD IT TO THE INDEX
                                19340           POP     D               ;POP OFF TEXT POINTER
                                19360           XCHG                    ;TEXT POINTER INTO (H,L)
                                19380           RET                     ;VARIABLE POINTER INTO (D,E)
                                19400
                                19420   NOTFDD: MOV     M,E             ;PUT LOOKS DOWN
                                19440           INX     H
                                19460           MOV     M,D
                                19480           INX     H
                                19500           LXI     D,$CODE+44      ;DEFAULT SIZE IS 10
                                19520           LDA     DIMFLG          ;ARE WE DIMENSIONING
                                19540           ORA     A
                                19560           JZ      NOTDIM
                                19580           POP     D               ;POP OFF INDEX
                                19600           PUSH    D               ;PUT INDEX BACK ON
                                19620           INX     D
                                19640           INX     D
                                19660           INX     D
                                19680           INX     D
                                19700   NOTDIM: PUSH    D
                                19720           MOV     M,E             ;PUT LENGTH DOWN
                                19740           INX     H
                                19760           MOV     M,D
                                19780           INX     H
                                19800           PUSH    H
                                19820           DAD     D
                                19840           CALL    REASOM          ;MAKE SURE WE'RE NOT RUNNING
                                19860                                   ;INTO THE STACK
                                19880           SHLD    STREND          ;SETUP NEW STORAGE END
                                19900           POP     D
                                19920   ZERIT2: DCX     H
                                19940           MVI     M,0
                                19960           COMPAR
                                19980           JNZ     ZERIT2
                                20000           JMP     MAKOFN          ;FINISH UP
                                20020
                                20040   PAGE
```



```
                                20060   SUBTTL  MULTIPLE DIMENSION CODE
                                20080
                                20100   IFN     MULDIM,<
                                20120   ;
                                20140   ; FORMAT OF ARRAYS IN CORE
                                20160   ;
                                20180   ; DESCRIPTOR
                                20200   ;       LOW BYTE = SECOND CHARACTER (200 BIT IS STRING FLAG)
                                20220   ;       HIGH BYTE = FIRST CHARACTER
                                20240   ; LENGTH OF ARRAY IN CORE IN BYTES (DOES NOT INCLUDE DESCRIPTOR)
                                20260   ; NUMBER OF DIMENSIONS 1 BYTE
                                20280   ; FOR EACH DIMENSION STARTING WITH THE FIRST A LIST
                                20300   ; (2 BYTES EACH) OF THE MAX INDICE+1
                                20320   ; THE VALUES
                                20340   ;
006745' 001000  000345          20360   ISARY1: PUSH    H               ;SAVE DIMFLG AND VALTYP FOR RECURSION
006746' 001000  000052          20380           LHLD    DIMFLG
006747' 000000  001542'
006750' 000000  006730'
006751' 001000  000343          20400           XTHL                    ;TEXT POINTER BACK INTO (H,L)
006752' 001000  000026          20420           MVI     D,0             ;SET # DIMENSIONS =0
006753' 000000  000000
006754' 001000  000325          20440   INDLOP: PUSH    D               ;SAVE NUMBER OF DIMENSIONS
006755' 001000  000305          20460           PUSH    B               ;SAVE LOOKS
006756' 001000  000315          20480           CALL    INTIOX          ;EVALUATE INDICE INTO (D,E)
006757' 000000  003622'
006760' 000000  006747'
006761' 001000  000301          20500           POP     B               ;POP OFF THE LOOKS
006762' 001000  000361          20520           POP     PSW             ;(A) = NUMBER OF DIMENSIONS SO FAR
006763' 001000  000353          20540           XCHG                    ;(D,E)=TEXT POINTER
                                20560                                   ;(H,L)=INDICE
006764' 001000  000343          20580           XTHL                    ;PUT THE INDICE ON THE STACK
                                20600                                   ;(H,L)=VALTYP & DIMFLG
006765' 001000  000345          20620           PUSH    H               ;RESAVE VALTYP AND DIMFLG
006766' 001000  000353          20640           XCHG                    ;(H,L)=TEXT POINTER
006767' 001000  000074          20660           INR     A               ;INCREMENT # OF DIMENSIONS
006770' 001000  000127          20680           MOV     D,A             ;(D)=NUMBER OF DIMENSIONS
006771' 001000  000176          20700           MOV     A,M             ;GET TERMINATING CHARACTER
006772' 001000  000376          20720           CPI     44              ;A COMMA SO MORE INDICES FOLLOW?
006773' 000000  000054
006774' 001000  000312          20740           JZ      INDLOP          ;IF SO, READ MORE
006775' 000000  006754'
006776' 000000  006757'
006777' 001000  000317          20760           SYNCHK  ")"             ;MAKE SURE IT ENDED PROPERLY
007000' 000000  000051
007001' 001000  000042          20780           SHLD    TEMP2           ;SAVE THE TEXT POINTER
007002' 000000  001605'
007003' 000000  006775'
007004' 001000  000341          20800           POP     H               ;(H,L)= VALTYP & DIMFLG
007005' 001000  000042          20820           SHLD    DIMFLG          ;SAVE VALTYP AND DIMFLG
007006' 000000  001542'
007007' 000000  007006'
007010' 001000  000325          20840           PUSH    D               ;SAVE NUMBER OF DIMENSIONS
```

```
                                        20860   ;
                                        20880   ; AT THIS POINT [B,C]=LOCKS, THE TEXT POINTER IS IN TEMP2,
                                        20900   ; THE INDICES ARE ALL ON THE STACK, FOLLOWED BY THE NUMBER OF DIMENSIONS,
                                        20920   ;
007611' 001000  000052  20940           LHLD    ARYTAB          ;[H,L]=PLACE TO START THE SEARCH
007612' 000000  001623'
007615' 000000  007065'
007618' 001000  000001  20960           XWD     ^D1000,^O76     ;"MVI A," AROUND THE NEXT BYTE
007619' 001000  000031  20980   LOPFOA: DAD     D               ;SKIP OVER THIS ARRAY SINCE IT'S
                        21000                                   ;NOT THE ONE
007616' 001000  000353  21020           XCHG                    ;[D,E]=CURRENT SEARCH POINT
007617' 001000  000052  21040           LHLD    STREND          ;GET THE PLACE TO STOP INTO [H,L]
007620' 000000  001625'
007621' 000000  007612'
007622' 001000  000353  21050           XCHG                    ;[H,L]=SEARCH POINT
007623' 001000  000347  21060           COMPAR                  ;STOPPING TIME?
                        21080   IFE     LENGTH-2,<
007624' 001000  000072  21100           LDA     VALTYP>
007625' 000000  001543'
007626' 000000  007020'
007627' 001000  000312  21120           JZ      NOTFOU          ;YES,COULDN'T FIND THIS ARRAY
007630' 000000  007104'
007631' 000000  007025'
                        21140   IFE     LENGTH-2,<
007632' 001000  000276  21160           CMP     M               ;SEE IF THE VALTYPS ARE THE SAME
007633' 001000  000043  21180           INX     H
007634' 001000  000302  21200           JNZ     NMANY2>
007635' 000000  007050'
007636' 007637' 001000  000176  21240           MOV     A,M             ;GET FIRST CHARACTER
007640' 001000  000071  21260           CMP     C               ;SEE IF IT MATCHES
007641' 001000  000043  21280           INX     H
007642' 001000  000302  21300           JNZ     NMANY1          ;NOT THIS ONE
007643' 000000  007051'
007644' 000000  007035'
007645' 001000  000176  21320           MOV     A,M             ;GET SECOND CHARACTER
007646' 001000  000270  21340           CMP     B               ;ANOTHER MATCH?
                        21360   IFE     LENGTH-2,<
007647' 001000  000076  21380           XWD     ^D1000,^O76     ;SKIP THIS INCREMENT WITH "MVI A,"
007650' 001000  000043  21400   NMANY2: INX     H>
007651' 001000  000043  21420   NMANY1: INX     H               ;POINT TO SIZE ENTRY
007652' 001000  000136  21440           MOV     E,M             ;[D,E]=LENGTH
007653' 001000  000043  21460           INX     H               ;OF THE ARRAY BEING LOOKED AT
007654' 001000  000126  21480           MOV     D,M
007655' 001000  000043  21500           INX     H
007656' 001000  000302  21520           JNZ     LOPFOA          ;IF NO MATCH, SKIP THIS ONE
007657' 000000  007015'
007660' 000000  007043'
                        21540                                   ;AND TRY AGAIN
007661' 001000  000072  21560           LDA     DIMFLG          ;SEE IF CALLED BY "DIM"
007664' 000000  001542'
007663' 000000  007057'
007664' 001000  000267  21580           ORA     A               ;ZERO MEANS NO
```



```
007665' 001000  000036  21600           MVI     E,ERRDD         ;"DOUBLY DIMENSIONED" ERROR
007666' 000000  000012
007667' 001000  000302  21620           JNZ     ERROR
007670' 000000  007102'
007671' 000000  007062'
                        21640   ;
                        21660   ; TEMP2=THE TEXT POINTER
                        21680   ; WE HAVE LOCATED THE VARIABLE WE WERE LOOKING FOR
                        21700   ; AT THIS POINT [H,L] POINTS BEYOND THE SIZE TO THE NUMBER OF DIMENSIONS
                        21720   ; THE INDICES ARE ON THE STACK FOLLOWED BY THE NUMBER OF DIMENSIONS
                        21740   ;
007674' 001000  000361  21760           POP     PSW             ;[A]=NUMBER OF DIMENSIONS
007675' 001000  000276  21780           CMP     M               ;MAKE SURE THE NUMBER GIVEN NOW AND
                        21800                                   ;AND WHEN THE ARRAY WAS SET UP ARE THE
                        21820                                   ;SAME
007674' 001000  000312  21840           JZ      GETOEP          ;JUMP OFF AND READ
007675' 000000  007645'
007676' 000000  007077'
                        21860                                   ;THE INDICES....
007677' 001000  000036  21880   BSERR:  MVI     E,ERRBS         ;"SUBSCRIPT OUT OF RANGE"
007700' 000000  000011
007701' 001000  000303  21900           JMP     ERROR
007702' 000000  002122'
007703' 000000  007075'
                        21920   ;
                        21940   ; HERE WHEN VARIABLE IS NOT FOUND IN THE ARRAY TABLE
                        21960   ;
                        21980   ; BUILDING AN ENTRY:
                        22000   ;
                        22020   ;       PUT DOWN THE DESCRIPTOR
                        22040   ;       SETUP NUMBER OF DIMENSIONS
                        22060   ;       MAKE SURE THERE IS ROOM FOR THE NEW ENTRY
                        22080   ;       REMEMBER VARPTR
                        22100   ;       TALLY=8 (VALTYP FOR THE EXTENDED)
                        22120   ;       SKIP 2 LOCS FOR LATER FILL IN == THE SIZE
                        22140   ; LOOP: GET AN INDICE
                        22160   ;       PUT NUMBER +1 DOWN AT VARPTR AND INCREMENT VARPTR
                        22180   ;       TALLY= TALLY + NUMBER+1
                        22200   ;       DECREMENT NUMBER=DIMS
                        22220   ;       JNZ     LOOP
                        22240   ;       CALL REASON WITH [H,L] REFLECTING LAST LOC OF VARIABLE
                        22260   ;       UPDATE STREND
                        22280   ;       ZERO BACKWARDS
                        22300   ;       MAKE TALLY INCLUDE MAXDIMS
                        22320   ;       PUT DOWN TALLY
                        22340   ;       IF CALLED BY DIMENSION, RETURN
                        22360   ;       OTHERWISE INDEX INTO THE VARIABLE AS IF IT
                        22380   ;       WERE FOUND ON THE INITIAL SEARCH
                        22400   ;
007704'                 22420   NOTFOU:
                        22440   IFE     LENGTH-2,<
007704' 001000  000167  22460           MOV     M,A             ;PUT DOWN THE VARIABLE TYPE
007705' 001000  000043  22480           INX     H
```

```
 6019  007106' 001000  000137          22500          MOV     E,A
 6020  007107' 001000  000026          22520          MVI     D,8>            ;[D,E]=SIZE OF ONE VALUE (VALTYP)
 6021  007110' 000000  000000
 6022                                  22540  IFN     LENGTH-2,<
 6023                                  22560          LXI     D,$CODE+4>      ;INITIALIZE TALLY TO FOUR
 6024  007111' 001000  000161          22580          MOV     M,C             ;PUT DOWN THE DESCRIPTOR
 6025  007112' 001000  000043          22600          INX     H
 6026  007113' 001000  000160          22620          MOV     M,B
 6027  007114' 001000  000043          22640          INX     H
 6028  007115' 001000  000361          22660          POP     PSW             ;[A]=NUMBER OF DIMENSIONS
 6029  007116' 001000  000062          22680          STA     TEMP6           ;SETUP GETSTK CALL
 6030  007117' 000000  007124'
 6031  007120' 000000  007102'
 6032  007121' 001000  000315          22700          CALL    GETSTK          ;GET SPACE FOR DIMENSION ENTRIES
 6033  007122' 000000  002024'
 6034  007123' 000000  007117'
 6035  007124' 001000  000351          22720  TEMP6:  PCHL                    ;PLACE TO STORE NUMBER OF DIMENSIONS
 6036                                  22740                                  ;FOR GETSTK AND LATER RECALL
 6037                                  22760                                  ;!!IMPURE!! PCHL TO CONFUSE DISASSEMBLY
 6038  007125' 001000  000042          22780          SHLD    TEMP3           ;SAVE THE LOCATION TO PUT THE SIZE
 6039  007126' 000000  001575'
 6040  007127' 000000  00712C'
 6041                                  22800                                  ;IN
 6042  007130' 001000  000043          22820          INX     H               ;SKIP OVER THE SIZE LOCATIONS
 6043  007131' 001000  000043          22840          INX     H
 6044  007132' 001000  000101          22860          MOV     B,C             ;[B]=NUMBER OF DIMENSIONS
 6045                                  22880                                  ;THIS DEPENDS ON THE FACT THAT GETSTK
 6046                                  22900                                  ;RETURNS ITS ARGUMENT IN [C]
 6047  007133' 001000  000160          22920          MOV     M,B             ;STORE THE NUMBER OF DIMENSIONS
 6048  007134' 001000  000043          22940          INX     H
 6049  007135' 001000  000072          22960  LOPPTA: LDA     DIMFLG          ;CALLED BY DIMENSION?
 6050  007136' 000000  001540'
 6051  007137' 000000  007126'
 6052  007140' 001000  000267          22980          ORA     A
 6053  007141' 001000  000170          23000          MOV     A,B             ;[A]=NUMBER OF DIMENSIONS
 6054  007142' 001000  000001          23020          LXI     B,$CODE+11      ;ASSUME IT'S NOT "DIM"
 6055  007143' 000000  000013'
 6056  007144' 000000  007136'
 6057  007145' 001000  000312          23040          JZ      NOTDIM          ;DEFAULT DIMENSIONS TO TEN
 6058  007146' 000000  007152'
 6059  007147' 000000  007143'
 6060  007150' 001000  000301          23060          POP     B               ;POP OFF AN INDICE INTO [B,C)
 6061  007151' 001000  000003          23080          INX     B               ;ADD ONE TO IT FOR THE ZERO ENTRY
 6062  007152' 001000  000161          23100  NOTDIM: MOV     M,C             ;PUT THE MAXIMUM DOWN
 6063  007153' 001000  000043          23120          INX     H
 6064  007154' 001000  000160          23140          MOV     M,B
 6065  007155' 001000  000043          23160          INX     H
 6066  007156' 001000  000365          23180          PUSH    PSW             ;SAVE THE NUMBER OF DIMENSIONS
 6067  007157' 001000  000345          23200          PUSH    H               ;SAVE THE POINTER INTO THE NEW ENTRY
 6068  007160' 001000  000315          23220          CALL    UMULT           ;MULTIPLY [B,C]=NEWMAX BY CURTOL=[D,E]
 6069  007161' 000000  002E00*
 6070  007162' 000000  007146'
 6071  007165' 001000  000353          23240          XCHG                    ;[D,E]=NEW CURTOL
```



```
 6072  007166' 001000  000341          23260          POP     H               ;GET THE POINTER INTO THE ENTRY BACK
 6073  007167' 001000  000301          23280          POP     B               ;GET THE NUMBER OF DIMENSIONS BACK
 6074  007168' 001000  000005          23300          DCR     B               ;DECREMENT THE NUMBER OF DIMENSIONS LEFT
 6075  007169' 001000  000302          23320          JNZ     LOPPTA          ;HANDLE THE OTHER INDICES
 6076  007170' 000000  007135'
 6077  007171' 000000  007161'
 6078  007172' 001000  000102          23340          MOV     B,D             ;[B,C)=SIZE
 6079  007173' 001000  000113          23360          MOV     C,E
 6080  007174' 001000  000353          23380          XCHG                    ;[D,E)=START OF VALUES
 6081  007175' 001000  000031          23400          DAD     D               ;[H,L]=END OF VALUES
 6082  007176' 001000  000332          23420          JC      BSENR           ;OUT OF MEMORY POINTER BEING GENERATED?
 6083  007177' 000000  007077'
 6084  007200' 000000  007170'
 6085  007201' 001000  000315          23440          CALL    REASON          ;SEE IF THERE IS ROOM FOR THE VALUES
 6086  007202' 000000  002045'
 6087  007205' 000000  007177'
 6088  007204' 001000  000042          23460          SHLD    STREND          ;UPDATE THE END OF STORAGE
 6089  007205' 000000  001625'
 6090  007206' 000000  007202'
 6091  007207' 001000  000053          23480  ZERITA: DCX     H               ;ZERO THE NEW ARRAY
 6092  007210' 001000  000066          23500          MVI     M,0
 6093  007211' 000000  000000
 6094  007212' 001000  000347          23520          COMPAR                  ;BACK AT THE BEGINNING?
 6095  007213' 001000  000302          23540          JNZ     ZERITA          ;NO, ZERO MORE
 6096  007214' 000000  007207'
 6097  007215' 000000  007205'
 6098  007216' 001000  000003          23560          INX     B               ;ADD ONE TO THE SIZE TO INCLUDE
 6099                                  23580                                  ;THE BYTE FOR THE NUMBER OF DIMENSIONS
 6100  007217' 001000  000147          23600          MOV     H,A             ;[H]=ZERO
 6101  007220' 001000  000072          23620          LDA     DIMFLG
 6102  007221' 000000  001542'
 6103  007222' 000000  007214'
 6104  007225' 001000  000267          23640          ORA     A               ;ARE WE DIMENSIONING ?
 6105  007224' 001000  000072          23660          LDA     TEMP6           ;GET THE NUMBER OF DIMENSIONS
 6106  007225' 000000  007124'
 6107  007226' 000000  007221'
 6108  007227' 001000  000157          23680          MOV     L,A             ;[L]=NUMBER OF DIMENSIONS
 6109  007230' 001000  000051          23700          DAD     H               ;[H,L]=NUMBER OF DIMENSIONS TIMES TWO
 6110  007231' 001000  000011          23720          DAD     B               ;ADD ON THE SIZE
 6111                                  23740                                  ;TO GET THE TOTAL NUMBER OF BYTES USED
 6112  007232' 001000  000353          23760          XCHG                    ;[D,E]=TOTAL SIZE
 6113  007233' 001000  000052          23780          LHLD    TEMP3           ;PLACE TO STORE SIZE
 6114  007234' 000000  001575'
 6115  007235' 000000  007225'
 6116  007236' 001000  000163          23800          MOV     M,E             ;PUT DOWN THE SIZE
 6117  007237' 001000  000043          23820          INX     H
 6118  007240' 001000  000162          23840          MOV     M,D
 6119  007241' 001000  000043          23860          INX     H
 6120  007242' 001000  000302          23880          JNZ     FINNOM
 6121  007245' 000000  007251'
 6122  007244' 000000  007234'
 6123                                  23900  ;
 6124                                  23920  ; AT THIS POINT [H,L] POINTS BEYOND THE SIZE TO THE NUMBER OF DIMENSIONS
```

```
                                        ;
                                        ; STRATEGY:
                                        ;       NUMDIM=NUMBER OF DIMENSIONS
                                        ;       CURTOL=0
                                        ; INLPNM:GET A NEW INDILE
                                        ;       POP NEW MAX INTO CURMAX
                                        ;       MAKE SURE INDICE IS NOT TOO BIG
                                        ;       MULTIPLY CURTOL BY CURMAX
                                        ;       ADD INDICE TO CURTOL
                                        ;       NUMDIM=NUMDIM-1
                                        ;       JNZ   INLPNM
                                        ;       USE CURTOL+4 (VALTYP FOR EXTENDED) AS OFFSET
                                        ;
007245' 001000  000043  24180 GETDEFI: INX   H                       ;POINT PAST THE NUMBER OF DIMENSIONS
007246' 001000  000001  24200         LXI   B,$CODE                  ;CURTOL=ZERO
007247' 000000  000100'
007250' 000000  007243'
007251' 001000  000026  24220         XWD   *01000,*026              ;*MVI D," AROUND THE NEXT BYTE
007252' 001000  000341  24240 INLPNM: POP   H                        ;(H,L)= POINTER INTO VARIABLE ENTRY
007253' 001000  000136  24260         MOV   E,M                      ;(D,E)=MAXIMUM FOR THE CURRENT INDICE
007254' 001000  000043  24280         INX   H
007255' 001000  000126  24300         MOV   D,M
007256' 001000  000043  24320         INX   H
007257' 001000  000343  24340         XTHL                           ;(H,L)=CURRENT INDILE
                                                                       ;POINTER INTO THE VARIABLE GOES ON THE STACK
007260' 001000  000365  24360         PUSH  PSW                      ;SAVE THE NUMBER OF DIMENSIONS
007261' 001000  000347  24400         COMPAR                         ;SEE IF THE CURRENT INDICE IS TOO BIG
007262' 001000  000322  24420         JNC   BSERR                    ;IF SO "BAD SUBSCRIPT" ERROR
007263' 000000  007077'
007264' 000000  007647'
007265' 001000  000345  24440         PUSH  H                        ;SAVE THE CURRENT INDICE
007266' 001000  000315  24460         CALL  UMULT                    ;CURTOL=CURTOL*CURRENT MAXIMUM
007267' 000000  007161*
007270' 000000  007663'
007271' 001000  000321  24480         POP   D                        ;INDICE INTO (D,E)
007272' 001000  000031  24500         DAD   D                        ;ADD THE INDICETO CURTOL
007273' 001000  000361  24520         POP   PSW                      ;GET THE NUMBER OF DIMENSIONS IN (A)
007274' 001000  000075  24540         DCR   A                        ;SEE IF ALL THE INDICES HAVE BEEN PROCESSED
007275' 001000  000104  24560         MOV   B,H                      ;(B,C)=CURTOL IN CASE WE LOOP BACK
007276' 001000  000115  24580         MOV   C,L
007277' 001000  000302  24600         JNZ   INLPNM                   ;PROCESS THE REST OF THE INDICES
007300' 000000  007252'
007301' 000000  007667'
                                        24620 IFE   LENGTH-2,<
007302' 001000  000072  24640         LDA   VALTYP                   ;SEE HOW BIG THE VALUES ARE
007303' 000000  001543'
007304' 000000  007300'
                                        24660                          ;AND MULTIPLY BY THAT SIZE
007305' 001000  000104  24680         MOV   B,M                      ;SAVE THE ORIGINAL VALUE FOR MULTIPLYING
007306' 001000  000115  24700         MOV   C,L                      ;BY THRLE
007307' 001000  000051  24720         DAD   H                        ;MULTIPLY BY TWO AT LEAST
007310' 001000  000326  24740         SUI   4                        ;FOR INTEGERS AND STRINGS
007311' 000000  000004
                                        24760                          ;NO MORE MULTIPLYING BY TWO
```



```
007312' 001000  000332  24780         JC    SMLVAL
007313' 000000  007522'
007314' 000000  007303'
007315' 001000  000051  24800         DAD   H                        ;NOW MULTIPLIED BY FOUR
007316' 001000  000312  24820         JZ    DONMUL                   ;IF SINGLE ALL DONE
007317' 000000  007326'
007320' 000000  007313'
007321' 001000  000051  24840         DAD   H                        ;BY EIGHT FOR DOUBLES
007322' 001000  000342  24860 SMLVAL: JPO   DONMUL                   ;FOR STRINGS
007323' 000000  007326'
007324' 000000  007317'
007325' 001000  000011  24880         DAD   B                        ;ADD IN THE ORIGINAL
007326'                 24900 DONMUL:>
                        24920 IFN   LENGTH-2,<
                        24940         DAD   H                        ;MULTIPLY CURTOL BY FOUR
                        24960         DAD   H>
007326' 001000  000301  24980         POP   B                        ;POP OFF THE ADDRESS OF WHERE THE VALUES
                        25000                                        ;BEGIN
007327' 001000  000011  25020         DAD   B                        ;ADD IT ONTO CURTOL TO GET THE
                        25040                                        ;PLACE THE VALUE IS STORED
007330' 001000  000353  25060         XCHG                           ;RETURN THE POINTER IN (D,E)
007331' 001000  000052  25080 FINNOW: LHLD  TEMP2                    ;REGET THE TEXT POINTER
007332' 000000  001605'
007333' 000000  007323'
007334' 001000  000053  25100         DCX   H                        ;REREAD THE TERMINATING CHARACTER
007335' 001000  020327  25120         CHRGET
007336' 001000  000311  25140         RET>
                        25160
                        25180 PAGE
```

```
                                        25200   SUBTTL FRE  FUNCTION AND INTEGER TO FLOATING  ROUTINES
                                        25220   IFN     LENGTH,<
 6209  007337' 001000  000056          25240   FRE:    LHLD    STREND          ;GET END OF VARIABLE AND TEXT SPACE
 6210  007340' 000000  001625'
 6211  007341' 000000  007336'
 6212  007342' 001000  000353          25260           XCHG                    ;PUT IT IN [D,E] FOR SUBTRACTION
 6213  007343' 001000  000041          25280           LXI     H,$CODE         ;ZERO [H,L]
 6214  007344' 000000  000000'
 6215  007345' 000000  007340'
 6216  007346' 001000  000071          25300           DAD     SP              ;PUT THE STACK POINTER IN [H,L]
 6217
 6218  ------
 6219  007347' 001000  000315          25320   IFN     STRING,<
 6220  007350' 000000  006407'          25340   IFE     LENGTH-2,<
 6221  007351' 000000  007344'          25360           CALL    GETYPE
 6222  007352' 001000  000302          25370           JNZ     GIVOBL>
 6223  007353' 000000  007576'
 6224  007354' 000000  007350'
 6225                                   25380   IFN     LENGTH-2,<
 6226                                   25400           LDA     VALTYP          ;WAS THE ARGUMENT A STRING?
 6227                                   25420           ORA     A
 6228                                   25440           JZ      GIVOBL>         ;NO, GIVE FREE VARIABLE SPACE
 6229  007355' 001000  000315          25460           CALL    FREFAC          ;FREE UP ARGUMENT AND SETUP
 6230  007356' 000000  010434'
 6231  007357' 000000  007353'
 6232                                   25480                                   ;TO GIVE FREE STRING SPACE
 6233  007360' 001000  000315          25500           CALL    GARBAG          ;DO GARBAGE COLLECTION
 6234  007361' 000000  010742'
 6235  007362' 000000  007356'
 6236  007363' 001000  000052          25520           LHLD    STKTOP          ;BOTTOM OF FREE AREA
 6237  007364' 000000  001613'
 6238  007365' 000000  007361'
 6239  007366' 001000  000353          25540           XCHG
 6240  007367' 001000  000052          25560           LHLD    FRETOP>>        ;TOP OF FREE AREA
 6241  007370' 000000  001573'
 6242  007371' 000000  007364'
 6243                                   25580   ;
 6244                                   25600   ; THIS ROUTINE SUBTRACTS [D,E] FROM [H,L]
 6245                                   25620   ; AND FLOATS THE RESULT LEAVING IT IN FAC.
 6246                                   25640   ;
 6247                                   25660   IFE     LENGTH-1,<
 6248                                   25680   GIVOBL: MOV     A,L             ;DO THE SUBTRACTION
 6249                                   25700           SUB     E
 6250                                   25720           MOV     C,A
 6251                                   25740           MOV     A,H
 6252                                   25760           SBB     D
 6253                                   25780   GIVACF: MOV     B,C>
 6254                                   25790   IFN     LENGTH-2,<
 6255                                   25800   GIVABF: MOV     D,B
 6256                                   25820           MVI     E,B             ;GET ZERO IN LOW
 6257                                   25840   IFN     STRING,<
 6258                                   25860           LXI     H,VALTYP        ;FLAG VALUE TYPE AS NUMERIC
 6259                                   25880           MOV     M,E>
```



```
                                        25900           MVI     B,144           ;SETUP TO FLOAT [B,C]
 6261                                   25920           JMP     FLOATX>
 6262                                   26120   IFE     LENGTH-2,<
 6263  007372' 001000  000175          26140   GIVOBL: MOV     A,L             ;[H,L]=[H,L]-[D,E]
 6264  007373' 001000  000023          26160           SUB     E
 6265  007374' 001000  000157          26180           MOV     L,A
 6266  007375' 001000  000174          26200           MOV     A,H
 6267  007376' 001000  000232          26220           SBB     D
 6268  007377' 001000  000021          26240           XWD     ^O1000,^O21     ;SKIP THE NEXT TWO BYTES WITH "LXI D,"
 6269  007400' 001000  000157          26260   SNGFLT: MOV     L,A             ;MAKE [A] AN UNSIGNED INTEGER
 6270  007401' 001000  000257          26280           XRA     A
 6271  007402' 001000  000147          26300   GIVINT: MOV     M,A
 6272  007403' 001000  000303          26320           JMP     MAKINT>
 6273  007404' 000000  006471'
 6274  007405' 000000  007376'
 6275                                   26322   IFN     LENGTH,<
 6276                                   26324   IFN     LPTSW,<
 6277                                   26326   LPOS:   LDA     LPTPOS
 6278                                   26328           JMP     SNGFLT>
 6279  007406' 001000  000072          26330   POS:    LDA     TTYPOS          ;GET TELETYPE POSITION
 6280  007407' 000000  000047'
 6281  007410' 000000  007404'
 6282                                   26332   IFN     LENGTH-2,<
 6283                                   26334   SNGFLT: MOV     B,A             ;RETURN FLOATING 1 BYTE
 6284                                   26336           XRA     A               ;UNSIGNED FROM A
 6285                                   26338           JMP     GIVABF>>        ;GIVING 0-255
 6286
 6287                                   26360   PAGE
```

```
                                        26380   SUBTTL  SIMPLE-USER-DEFINED-FUNCTION CODE
                                        26390   IFN     FUNCTS,<
                                        26400   ;
                                        26420   ; NOTE ONLY SINGLE ARGUMENTS ARE ALLOWED TO FUNCTIONS
                                        26440   ; AND FUNCTIONS MUST BE OF THE SINGLE LINE FORM:
                                        26460   ; DEF FNA(X)=X^2+X-2
                                        26480   ; NO STRINGS CAN BE INVOLVED WITH THESE FUNCTIONS
                                        26500
                                        26520   ; IDEA: CREATE A FUNNY SIMPLE VARIABLE ENTRY
                                        26540   ; WHOSE FIRST CHARACTER (SECOND WORD IN MEMORY)
                                        26560   ; HAS THE 200 BIT SET.
                                        26580   ; THE VALUE WILL BE:
                                        26600   ;
                                        26620   ;       A TXTPTR TO THE FORMULA
                                        26640   ;       A PTR TO THE ARGUMENT VARIABLE
                                        26660   ;
                                        26680   ; FUNCTION NAMES CAN BE LIKE "FNA#"
                                        26700   ;

0307 007411' 021000  000315    26780   DEF:    CALL    GETFNM          ;GET A POINTER TO THE
0308 007412' 000000  007550'
0309 007415' 000000  007407'
                                        26800                           ;FUNCTION VARIABLE
0311 007414' 021000  000001    26820           LXI     B,DATA          ;EVENTUALLY RETURN TO "DATA"
0312 007415' 000000  004072'
0313 007418' 000000  007416'
0314 007417' 021000  000305    26840           PUSH    B               ;AND SKIP THE FORMULA
0315 007420' 021000  000525    26860           PUSH    D               ;SAVE A POINTER TO IT
0316 007421' 021000  000315    26880           CALL    ERROR           ;DEF IS "ILLEGAL DIRECT"
0317 007422' 000000  007532'
0318 007423' 000000  007415'
0319 007424' 021000  000317    26900           SYNCHK  "("             ;MUST HAVE "("
0320 007425' 000000  000050
                                        26920                           ;SINCE WE STORE A TEXT POINTER
0322 007426' 021000  000315    26940           CALL    PTRGET          ;GET POINTER TO ARGUMENT
0323 007427' 000000  006565'
0324 007430' 000000  007422'
                                        26960   IFN     LENGTH-2,<
                                        26980   IFN     STRING,<CALL    CHKNUM>>;STRINGS ILLEGAL
0327 007431' 021000  000317    27000           SYNCHK  ")"             ;MUST CLOSE IT WITH ")"
0328 007432' 000000  000051
0329 007433' 021000  000317    27020           SYNCHK  EQULTK          ;MUST HAVE EQUAL
0330 007434' 000000  000060
0331 007435' 021000  000104    27040           MOV     B,H
0332 007436' 021000  000115    27060           MOV     C,L
0333 007437' 021000  000343    27080           XTHL                    ;PUT THE TXTPTR ON THE STACK
                                        27100                           ;(H,L)=PTR TO FUNCTION VARIABLE
                                        27120                           ;(B,C)=TXTPTR
0336 007440' 021000  000303    27140           JMP     DEFFIN          ;PUT DOWN THE TEXT-POINTER
0337 007441' 000000  007521'
0338 007442' 000000  007421'
                                        27160                           ;AND ARGUMENT POINTER IN
0340                                    27180                           ;MEMORY, RESTORE THE TXTPTR
```



```
                                        27200                           ;AND GO TO "DATA" SKIPPING THE
                                        27220                           ;REST OF THE FORMULA

0344 007443' 021000  000315    27260   FNDDEF: CALL    GETFNM          ;GET A POINTER TO
0345 007444' 000000  007550'
0346 007445' 000000  007441'
                                        27280                           ;THE FUNCTION DEFINITION IN (D,E)
0348 007446' 021000  000325    27300           PUSH    D               ;SAVE THE POINTER
0349 007447' 021000  000315    27320           CALL    PARCHK          ;EVALUATE THE VALUE TO BE PASSED
0350 007450' 000000  006136'
0351 007451' 000000  007444'
                                        27340   IFN     LENGTH-2,<
                                        27360   IFN     STRING,<CALL    CHKNUM>> ;ARG CANNOT BE STRING
0354 007452' 021000  000343    27380           XTHL                    ;(H,L)=POINTER TO FUNCTION DEF
                                        27400                           ;TEXT POINTER GOES ON THE STACK
0356 007453' 021000  000367    27420           PUSHM                   ;PUSH THE POINTER AT THE FORMULA
                                        27440                           ;ONTO THE STACK
0358 007454' 021000  000321    27460           POP     D               ;(D,E)=PTR TO FORMULA
0359 007455' 021000  000367    27480           PUSHM                   ;PUT A POINTER TO THE
                                        27500                           ;ARGUMENT ON THE STACK
0361 007456' 021000  000341    27520           POP     H               ;(H,L)=POINTER TO ARG
0362 007457' 021000  000367    27540           PUSHFN                  ;SAVE ARGS OLD VALUE ON THE STACK
0363 007460' 021000  000367
0364 007461' 021000  000053    27560           DCX     H
0365 007462' 021000  000053    27580           DCX     H
0366 007463' 021000  000053    27600           DCX     H               ;POINT TO FRONT OF ARG AGAIN
0367 007464' 021000  000053    27620           DCX     H
0368 007465' 021000  000345    27640           PUSH    H               ;SAVE IT
0369 007466' 021000  000347    27660           COMPAR                  ;SHOULDN'T BE EQUAL UNLESS
                                        27680                           ;FUNCTION WAS NEVER DEFINED
0371 007467' 021000  000325    27700           PUSH    D               ;SAVE FORMULA TEXT POINTER
0372 007470' 021000  000036    27720           MVI     E,ERRLF         ;NOW LO,E) FREE SO CHECK IF (ZERO) SET
0373 007471' 000000  000022
0374 007472' 021000  000312    27740           JZ      ERROR
0375 007473' 000000  002122'
0376 007474' 000000  007450'
0377 007475' 021000  000315    27760           CALL    MOVMF           ;PUT CURRENT FAC INTO OUR ARG VARIABLE
0378 007476' 000000  005264'
0379 007477' 000000  007473'
                                        27780                           ;OUT OF FAC INTO (H,L) LOCATION
0381 007500' 021000  000341    27800           POP     H               ;POP OFF FORMULA TXTPTR
                                        27810   IFN     LENGTH-2,<
                                        27820           CALL    FRMNUM>         ;EVALUATE IT AND MUST BE ITS NUMERIC
                                        27822   IFE     LENGTH-2,<
0385 007501' 021000  000315    27824           CALL    FRMEVL
0386 007502' 000000  005336'
0387 007503' 000000  007476'
0388 007504' 021000  000345    27826           PUSH    H
0389 007505' 021000  000315    27828           CALL    FRCSNG
0390 007506' 000000  006677'
0391 007507' 000000  007502'
0392 007510' 021000  000341    27830           POP     H>
0393 007511' 021000  000053    27840           DCX     H
```

```
6394  007512' 001000  000327          27850          CMRGET                  ;SEE IF TERMINATED
6395  007515' 021000  000322          27860          JNZ     $NERR           ;IF NOT SYNTAX ERROR
6396  007518' 000000  002072'
6397  007519' 000000  007506'
6398                                   27900                                  ;TO BE NICE SHOULD HAVE NEW CURLIN
6399                                   27920                                  ;BUT VERY MESSY
6400  007516' 001002  000341          27940          POP     H               ;POP OFF POINTER AT ARG VARIABLE
6401  007517' 001002  000321          27960          POP     D
6402  007520' 001002  000301          27980          POP     B>              ;POP OFF OLD VALUE
6403                                   28000  IFN     MULJIM,STRING,FUNCTS,<
6404  007521' 001002  000161          28020  DEFFIN: MOV     M,C
6405  007522' 001002  000243          28040          INX     H               ;STORE THE OLD VALUE
6406  007523' 001002  000160          28060          MOV     M,B
6407  007524' 001002  000043          28080  PUTJEI: INX     H
6408  007525' 001002  000163          28100          MOV     M,E
6409  007526' 001002  000043          28120          INX     H
6410  007527' 001002  000162          28140          MOV     M,D
6411  007530' 001002  000341          28160          POP     H               ;POP OFF OLD TXTPTR
6412  007531' 001002  000311          28180          RET>                    ;VALUE IS IN FAC => ALL DONE
6413                                   28200  IFN     FUNCTS,<
6414                                   28220  ;
6415                                   28240  ; SUBROUTINE TO SEE IF WE ARE IN DIRECT MODE AND
6416                                   28260  ; COMPLAIN IF SO
6417                                   28280  ;
6418  007532' 001002  000345          28300  ERRDIR: PUSH    H               ;SAVE THEIR (H,L)
6419  007533' 001002  000052          28320          LHLD    CURLIN          ;SEE WHAT THE CURRENT LINE IS
6420  007534' 000000  001607'
6421  007535' 000000  007514'
6422  007536' 001002  000043          28340          INX     H               ;DIRECT IS 65,535 SO NOW 0
6423  007537' 001002  000174          28360          MOV     A,H
6424  007540' 001002  000265          28380          ORA     L               ;IS IT ZERO NOW?
6425  007541' 001002  000341          28400          POP     H
6426  007542' 001002  000300          28420          RNZ                     ;RETURN IF NOT
6427  007543' 001002  000036          28440          MVI     E,ERRIJ         ;"ILLEGAL DIRECT" ERROR
6428  007544' 000000  000014
6429  007545' 001000  000303          28460          JMP     ERROR
6430  007546' 000000  002102'
6431  007547' 000000  007534'
6432                                   28480  ;
6433                                   28500  ; SUBROUTINE TO GET A POINTER TO A FUNCTION NAME
6434                                   28520  ;
6435  007550' 001000  000317          28540  GETFNM: SYNCHK  FNTK            ;MUST START WITH "FN"
6436  007551' 000000  000245
6437  007552' 001000  000076          28560          MVI     A,128           ;DONT ALLOW AN ARRAY
6438  007553' 000000  000200
6439  007554' 001002  000062          28580          STA     SUBFLG          ;DON'T RECOGNIZE THE "(" AS
6440  007555' 000000  001601'
6441  007556' 000000  007546'
6442                                   28600                                  ;THE START OF AN ARRAY REFERENCE
6443  007557' 001000  000266          28620          ORA     M               ;PUT FUNCTION BIT ON
6444  007560' 001002  002107          28640          MOV     B,A             ;SET FIRST CHARACTER INTO [B]
6445                                   28660  IFN     LENGTH-2,<
6446                                   28680  IFN     STRING,<CALL    PTRGT2  ;REALLY GET THE POINTER
```



```
6447                                   28700          JMP     CMKNUM>>        ;MAKE SURE ITS NOT A STRING NAME
6448  007561' 001000  000303          28720  IFE     STRING&<LENGTH-2>,<JMP  PTRGT2>>
6449  007562' 000000  006516'
6450  007563' 000000  007555'
6451                                   28740  PAGE
```

```
                                        28760   SUBTTL  STRING FUNCTIONS
                                        28780   IFN     STRING,<                ;STRING HANDLING SUBROUTINES
                                        28800   ;
                                        28820   ; THE STR$ FUNCTION TAKES A NUMBER AND GIVES
                                        28840   ; A STRING WITH THE CHARACTERS THE OUTPUT OF THE NUMBER
                                        28860   ; WOULD HAVE GIVEN
                                        28880   ;
                                        28900   STR$:
                                        28920   IFN     LENGTH-2,<
                                        28940           CALL    CHKNUM          ;MAKE SURE THE ARGUMENT
                                        28960                                   ;IS A NUMERIC
                                        28980           CALL    FOUT            ;DO ITS OUTPUT
                                        29000           CALL    STRLIT          ;SCAN IT AND TURN IT INTO A STRING
                                        29020           CALL    FREFAC          ;FREE UP THE TEMP
                                        29040           LXI     B,FINBCK
                                        29060           PUSH    B               ;SET UP ANSWER IN NEW TEMP
                                        29080   ;
                                        29100   ; STRCPY CREATES A COPY OF THE STRING
                                        29120   ; WHOSE DESCRIPTOR IS POINTED TO BY [M,L].
                                        29140   ; ON RETURN [D,E] POINTS TO DSCTMP
                                        29160   ; WHICH HAS THE STRING INFO (LENGTH,WHERE COPIED TO)
                                        29180   ;
                                        29200   STRCPY: MOV     A,M             ;GET LENGTH
                                        29220           INX     H               ;MOVE UP TO THE POINTER
                                        29240   IFN     LENGTH-2,<
                                        29260           INX     H>
                                        29280           PUSH    H               ;GET POINTER TO POINTER OF ARG
                                        29300           CALL    GETSPA          ;GET THE SPACE
                                        29320           POP     H               ;FIND OUT WHERE STRING TO COPY
                                        29340           PUSH    H               ;REALLY IS IN [B,C]
                                        29360           POP     B
                                        29380           CALL    STRAD2          ;SETUP DSCTMP
                                        29400           PUSH    H               ;SAVE POINTER TO DSCTMP
                                        29420           MOV     L,A             ;GET CHARACTER COUNT INTO [L]
                                        29440           CALL    MOVSTR          ;MOVE THE CHARS IN
                                        29460           POP     D               ;RESTORE POINTER TO DSCTMP
                                        29480           RET                     ;RETURN
                                        29500   ;
                                        29520   STRINI: CALL    GETSPA          ;GET SOME STRING SPACE [A] CHARS]
```



```
                                        29540   STRAD2: LXI     M,DSCTMP        ;GET DESC. TEMP
                                        29560   STRAD1: PUSH    H               ;SAVE DESC. POINTER
                                        29580           MOV     M,A             ;SAVE CHARACTER COUNT
                                        29600   IFN     LENGTH-2,<
                                        29620           INX     H>              ;MOVE TO ADDRESS FIELD
                                        29640           JMP     PUTDE1          ;USE COMMON CODE TO
                                        29660                                   ;STORE [D,E]=POINTER TO FREE SPACE
                                        29680                                   ;AND RESTORE [H,L] AS THE DESCRIPTOR POINTER
                                        29682   ;
                                        29684   ; STRLIT TAKES THE STRING LITERAL WHOSE FIRST CHARACTER
                                        29686   ; IS POINTED BY [H,L]+1 AND BUILDS A DESCRIPTOR FOR IT,
                                        29688   ; THE DESCRIPTOR IS INITIALLY BUILT IN DSCTMP, BUT PUTNEW
                                        29690   ; TRANSFERS IT INTO A TEMPORARY AND LEAVES A POINTER
                                        29692   ; AT THE TEMPORARY IN FACLO. THE CHARACTERS OTHER THAN
                                        29694   ; ZERO THAT TERMINATE THE STRING SHOULD BE SET UP IN [B]
                                        29696   ; AND [D]. IT THE TERMINATOR IS A QUOTE, THE QUOTE IS SKIPPED
                                        29697   ; OVER. LEADING QUOTES SHOULD BE SKIPPED BEFORE CALL. ON RETURN
                                        29698   ; THE CHARACTER AFTER THE STRING LITERAL IS POINTED TO
                                        29700   ; BY [H,L] AND IS IN [A], BUT THE CONDITION CODES ARE
                                        29702   ; NOT SET UP.
                                        29704   ;
                                        29720   STRLIT: DCX     H
                                        29740           MVI     B,34            ;ASSUME STR ENDS ON QUOTE
                                        29760   STRLT3: MOV     D,B
                                        29780   STRLT2: PUSH    H               ;SAVE POINTER TO START OF LITERAL
                                        29800           MVI     C,255           ;INITIALIZE CHARACTER COUNT
                                        29820   STNGET: INX     H
                                        29840           MOV     A,M             ;GET CHAR
                                        29860           INR     C               ;BUMP CHARACTER COUNT
                                        29880           DRA     A               ;IF 0, (END OF LINE) DONE
                                        29900           JZ      STRFIN          ;TEST
                                        29920           CMP     D
                                        29940           JZ      STRFIN
                                        29960           CMP     B               ;CLOSING QUOTE
                                        29980           JNZ     STNGET          ;NO, GO BACK FOR MORE
                                        30000   STRFIN: CPI     34              ;IF QUOTE TERMINATES THE STRING
                                        30020           JZ      CHRGTR          ;SKIP OVER THE QUOTE
```

```
                                30040           XTHL
                                30050           INX     H
                                30060           XCHG                    ;GET POINTER TO TEMP
                                30070           MOV     A,C             ;GET CHARACTER COUNT IN A
                                30080           CALL    STRAD2          ;SAVE STR INFO
                                30090
                                30100                                   ;COMPARE DESC. POINTER & STK POINTER
                                30110           COMPAR                  ;TO SEE IF STR IN BUF
                                30120           CNC     STRCPY          ;IF IN BUF, MUST COPY IT.
                                30130
                                30140                                   ;OTHERWISE STR IN PROGRAM.
                                30150   ;
                                30160   ; SOME STRING FUNCTION IS RETURNING A RESULT IN DSCTMP
                                30170   ; WE WANT TO SETUP A TEMP DESCRIPTOR WITH DCSTMP IN IT
                                30180   ; PUT A POINTER TO THE DESCRIPTOR IN FACLO AND FLAG THE
                                30190   ; RESULT AS TYPE STRING
                                30200   ;
                                30210   PUTNEW: LXI     O,DSCTMP        ;[D,E) POINT AT RESULT DESCRIPTOR
                                30220
                                30230           LHLD    TEMPPT          ;[H,L]=POINTER TO FIRST FREE TEMP
                                30240
                                30250           SHLD    FACLO           ;POINTER AT WHERE RESULT DESCRIPTOR WILL BE
                                30260
                                30270           IFE     LENGTH-2,<
                                30280           MVI     A,3
                                30290           STA     VALTYP          ;FLAG THIS AS A STRING
                                30300
                                30310           CALL    VMOVE>          ;AND MOVE THE VALUE INTO A TEMPORARY
                                30320
                                30330           IFN     LENGTH-2,<
                                30340           MVI     A,1             ;FLAG RESULT AS STRING
                                30350           STA     VALTYP          ;STORE IN "VALUE-TYPE" LOCATION
                                30360           CALL    MOVE>           ;TRANSFER THE DESCRIPTOR IN
                                30370           COMPAR                  ;DSCTMP IS JUST BEYOND THE TEMPS
                                30380                                   ;AND IF TEMPPT POINTS AT IT THERE
                                30390                                   ;ARE NO FREE TEMPS
                                30400           MVI     E,ERRST         ;"STRING TEMPORARY" ERROR
                                30410           JZ      ERROR           ;GO TELL HIM
                                30420
                                30430           SHLD    TEMPPT          ;SAVE NEW TEMPORARY POINTER
```



```
                                30440           POP     H               ;GET THE TEXT POINTER
                                30450           MOV     A,M             ;GET CURRENT CHARACTER INTO [A]
                                30460           RET
                                30470   ;
                                30480   ; PRINT THE STRING POINTED TO BY [H,L] WHICH ENDS WITH A ZERO
                                30490   ; IF THE STRING IS BELOW DSCTMP IT WILL BE COPIED INTO STRING SPACE
                                30500   ;
                                30510   STROUT: INX     H               ;POINT AT NEXT CHARACTER
                                30520           CALL    STRLIT          ;GET A STRING LITERAL
                                30530
                                30540   ;
                                30550   ; PRINT THE STRING WHOSE DESCRIPTOR IS POINTED TO BY FACLO.
                                30560   ;
                                30570   STRPRT: CALL    FREFAC          ;RETURN TEMP POINTER BY FACLO
                                30580
                                30590           IFE     LENGTH-2,<
                                30600           CALL    GETSCD          ;[D]=LENGTH [B,C]=POINTER AT DATA
                                30610
                                30620           INR     D               ;INCREMENT AND DECREMENT EARLY
                                30630                                   ;TO CHECK FOR NULL STRING
                                30640   STRPR2: DCR     D>              ;DECREMENT THE LENGTH
                                30650           IFN     LENGTH-2,<
                                30660           CALL    MOVRM           ;GET LENGTH IN [E]
                                30670           INR     E               ;AND POINTER TO CHARACTERS IN [B,C]
                                30680   STRPR2: DCR     E>
                                30690           RZ                      ;ALL DONE
                                30700           LDAX    B               ;GET CHARACTER TO PRINT
                                30710           OUTCHR
                                30720           CPI     CR
                                30730
                                30740           CZ      CRFIN
                                30750
                                30760           INX     B               ;POINT TO THE NEXT CHARACTER
                                30770           JMP     STRPR2          ;AND PRINT IT...
                                30780
                                30790   ;
                                30800   ; GETSPA = GET SPACE FOR CHARACTER STRING
                                30810   ; MAY FORCE GARBAGE COLLECTION.
                                30820   ;
                                30830   ; # OF CHARS (BYTES) IN [A]
                                30840   ; RETURNS WITH POINTER IN [D,E] OTHERWISE IF CANT GET SPACE
                                30850   ; BLOWS OFF TO "OUT OF STRING SPACE" TYPE ERROR.
                                30860   ;
                                30870   GETSPA: ORA     A               ;MUST BE NON ZERO. SIGNAL NO GARBAG YET
                                30880           XWD     *D1000,*0016    ;*MVI C" AROUND THE NEXT
                                30890   THYGI2: POP     PSW             ;IN CASE COLLECTED WHAT WAS LENGTH?
                                30900           PUSH    PSW             ;SAVE IT BACK
                                30910           LHLD    STKTOP          ;GET BOTTOM OF STRING SPACE
```

```
6664  007777' 000000  001615'
6665  010000' 000000  007770'
6666  010001' 001000  000353          31540           XCHG                    ;IN (D,E)
6667  010002' 001000  000052          31560           LHLD    FRETOP          ;GET TOP OF FREE SPACE IN (H,L)
6668  010005' 000000  001573'
6669  010004' 000000  007777'
6670  010005' 001000  000057          31580           CMA                     ;-# OF CHARS
6671  010006' 001000  000117          31600           MOV     C,A             ;IN (B,C)
6672  010007' 001000  000026          31620           MVI     B,255
6673  010010' 000000  000377
6674  010011' 001000  000011          31640           DAD     B
6675  010012' 001000  000043          31660           INX     H
6676  010013' 001000  000347          31680           COMPAR
6677  010014' 001000  000332          31700           JC      GARBAG          ;NOT ENOUGH ROOM FOR STRING, OFFAL TIME
6678  010015' 000000  010026'
6679  010016' 000000  010003'
6680  010017' 001000  000042          31720           SHLD    FRETOP          ;SAVE NEW BOTTOM OF MEMORY
6681  010020' 000000  001573'
6682  010021' 000000  010015'
6683  010022' 001000  000043          31740           INX     H               ;MOVE BACK TO POINT TO STRING
6684  010023' 001000  000353          31760           XCHG                    ;RETURN WITH POINTER IN (D,E)
6685  010024' 001000  000361          31780   PPSWRT: POP     PSW             ;GET CHARACTER COUNT
6686  010025' 001000  000311          31800           RET                     ;RETURN FROM GETSPA
6687
6688  010026' 001000  000361          31820   GARBAG: POP     PSW             ;HAVE WE COLLECTED BEFORE?
6689  010027' 001000  000036          31840           MVI     E,ERRSO         ;GET READY FOR OUT OF STRING SPACE ERROR
6690  010030' 000000  000016
6691  010031' 001000  000312          31860           JZ      ERROR           ;GO TELL USER HE LOST
6692  010032' 000000  002106'
6693  010035' 000000  010020'
6694  010034' 001000  000277          31880           CMP     A               ;SET ZERO FLAG TO SAY WE'VE GARBAGED
6695  010035' 001000  000365          31900           PUSH    PSW             ;SAVE FLAG BACK ON STACK
6696  010036' 001000  000001          31920           LXI     B,TRYGI2        ;PLACE FOR GARBAG TO RETURN TO.
6697  010037' 000000  007774'
6698  010040' 000000  010042'
6699  010041' 001000  000325          31940           PUSH    B               ;SAVE ON STACK
6700  010042' 001000  000052          31960   GARBA2: LHLD    MEMSIZ          ;START FROM TOP DOWN
6701  010045' 000000  001545'
6702  010044' 000000  010037'
6703                                   32000   IFE     READIO,<
6704                                   32020           MVI     A,7             ;RING THE BELL ON GARBAGE COLLECTION
6705                                   32040           OUTCHR>
6706  010045' 001000  000042          32060   FNDVAR: SHLD    FRETOP          ;LIKE SO
6707  010046' 000000  001573'
6708  010047' 000000  010043'
6709  010050' 001000  000041          32080           LXI     H,$COLE         ;GET DOUBLE ZERO
6710  010051' 000000  010061'
6711  010052' 000000  010040'
6712  010053' 001000  000345          32100           PUSH    H               ;SAY DIDNT SEE VARS THIS PASS
6713  010054' 001000  000052          32120           LHLD    STKTOP          ;FORCE DVARS TO IGNORE STRINGS
6714  010055' 000000  001615'
6715  010056' 000000  010051'
6716                                   32140                                   ;IN THE PROGRAM TEXT (LITERALS, DATA)
```



```
6717  010057' 001000  000345          32160           PUSH    H               ;FORCE FIND HIGH ADDRESS
6718  010060' 001000  000041          32180           LXI     H,TEMPST        ;GET START OF STRING TEMPS
6719  010061' 000000  001551'
6720  010062' 000000  010055'
6721  010065' 001000  000353          32200   TVAR:   XCHG                    ;SAVE IN (D,E)
6722  010064' 001000  000052          32220           LHLD    TEMPPT          ;SEE IF DONE
6723  010065' 000000  001547'
6724  010066' 000000  010061'
6725  010067' 001000  000353          32240           XCHG                    ;FLIP
6726  010070' 001000  000347          32260           COMPAR                  ;TEST
6727  010071' 001000  000301          32280           LXI     B,TVAR          ;FORCE JUMP TO TVAR
6728  010072' 000000  010063'
6729  010073' 000000  010065'
6730  010074' 001000  000302          32300           JNZ     DVAR2           ;DO TEMP VAR GARBAGE COLLECT
6731  010075' 000000  010213'
6732  010076' 000000  010072'
6733
6734  010077' 001000  000052          32320   SVARS:  LHLD    VARTAB          ;GET START OF SIMPLE VARIABLES
6735  010100' 000000  001621'
6736  010101' 000000  010075'
6737  010102' 001000  000353          32340   SVAR:   XCHG                    ;GET IN (D,E)
6738  010103' 001000  000052          32360           LHLD    ANYTAB          ;GET END OF SIMPS
6739  010104' 000000  001623'
6740  010105' 000000  010100'
6741  010106' 001000  000353          32380           XCHG                    ;FLIP
6742  010107' 001000  000347          32400           COMPAR                  ;SEE IF AT END OF SIMPS
6743  010110' 001000  000312          32420           JZ      ARYVAR          ;IF YES, DO ARRAY TYPE STRINGS
6744  010111' 000000  010140'
6745  010112' 000000  010104'
6746  010113' 001000  000176          32440           MOV     A,M             ;GET 2ND CHARACTER OF VARIABLE
6747  010114' 001000  000043          32460           INX     H               ;BUMP POINTER TWICE
6748  010115' 001000  000043          32480           INX     H               ;
6749                                   32500   IFE     LENGTH-2,<
6750  010116' 001000  000043          32520           INX     H               ;POINT AT THE VALUE
6751  010117' 001000  000376          32540           CPI     3               ;SEE IF ITS A STRING
6752  010120' 000000  000003
6753  010121' 001000  000302          32560           JNZ     SKPVAR          ;IF NOT, JUST SKIP AROUND IT
6754  010122' 000000  010130'
6755  010123' 000000  010111'
6756  010124' 001000  000315          32580           CALL    DVARS           ;COLLECT IT
6757  010125' 000000  010214'
6758  010126' 000000  010122'
6759  010127' 001000  000257          32620           XRA     A               ;AND DON'T SKIP ANYTHING MORE
6760  010130' 001000  000137          32640   SKPVAR: MOV     E,A
6761  010131' 001000  000026          32660           MVI     D,0             ;(D,E)=AMOUNT TO SKIP
6762  010132' 000000  000000
6763  010133' 001000  000031          32680           DAD     D>
6764                                   32700   IFN     LENGTH-2,<
6765                                   32720           ORA     A               ;SET CC'S
6766                                   32740           CALL    DVARS>          ;CALL THE VARIABLE GARB ROUT.
6767  010134' 001000  000303          32760           JMP     SVAR            ;GET NEXT ONE
6768  010135' 000000  010102'
6769  010136' 000000  010125'
```

```
                                                                
                                                                
0101371' 001000  000301          32800   ARYVA2: POP     B               ;GET RID OF STACK GARBAGE
0101401' 001000  000353          32820   ARYVAR: XCHG                    ;SAVE ARYVAR IN (D,E)
0101411' 001000  000052          32840           LHLD    STREND          ;GET END OF ARRAYS
0101422' 000000  001625'
0101451' 000000  010135'
0101441' 001000  000353          32860           XCHG                    ;FLIP BACK
0101451' 001000  000347          32880           COMPAR                  ;SEE IF DONE WITH ARRAYS
0101461' 001000  000312          32900           JZ      GWBPAS          ;YES, SEE IF DONE COLLECTING
0101471' 000000  010254'
0101501' 000000  010146'
                                 32920   IFE     LENGTH=2,<
0101511' 001000  000176          32940           MOV     A,M             ;GET THE VALUE TYPE INTO (A)
0101521' 001000  000043          32960           INX     H>
0101531' 001000  000315          32980           CALL    MOVRM           ;GET LENGTH OF ARRAY IN (B,C)
0101541' 000000  005302*
0101551' 000000  010147'
                                 33000   IFN     LENGTH=2,<
                                 33020           MOV     A,E>            ;GET 2ND CHAR OF VAR NAME IN A
0101561' 001000  000345          33040           PUSH    H               ;SAVE POINTER TO DIMS
0101571' 001000  000011          33060           DAD     B               ;ADD TO CURRENT POINTER POSITION
                                 33080   IFE     LENGTH=2,<
0101601' 001000  000376          33100           CPI     '$              ;SEE IF ITS A STRING
0101611' 000000  000003
0101621' 001000  000302          33120           JNZ     ARYVA2>         ;IF NOT JUST SKIP IT
0101631' 000000  010137'
0101641' 000000  010154'
                                 33140   IFN     LENGTH=2,<
                                 33160           ORA     A               ;SEE IF STRING VAR
                                 33180           JP      ARYVA2>         ;NO, KEEP ON TRUCKIN
0101651' 000000  000042          33200           SHLD    TEMP3           ;SAVE END OF ARRAY
0101661' 000000  001575'
0101671' 000000  010163'
0101701' 001000  000341          33220           POP     H               ;GET BACK CURRENT POSITION
0101711' 001000  000116          33240           MOV     C,M             ;PICK UP NUMBER OF DIMS
0101721' 001000  000006          33260           MVI     B,0             ;MAKE DOUBLE WITH HIGH ZERO
0101731' 000000  000000
0101741' 001000  000011          33280           DAD     B               ;GO PAST DIMS
0101751' 001000  000011          33300           DAD     B               ;BY ADDING ON TWICE #DIMS (2 BYTE GUYS)
0101761' 001000  000043          33320           INX     H               ;ONE MORE TO ACCOUNT FOR #DIMS.
0101771' 001000  000353          33340   ARYSTR: XCHG                    ;SAVE CURRENT POSIT IN (D,E)
0102001' 001000  000052          33360           LHLD    TEMP3           ;GET END OF ARRAY
0102011' 000000  001575'
0102021' 000000  010166'
0102031' 001000  000355          33380           XCHG                    ;FIX (H,L) BACK TO CURRENT
0102041' 001000  000347          33400           COMPAR                  ;SEE IF AT END OF ARRAY
0102051' 001000  000312          33420           JZ      ARYVAR          ;END OF ARRAY, TRY NEXT ARRAY
0102061' 000000  010140'
0102071' 000000  010201'
0102101' 001000  000001          33440           LXI     B,ARYSTR        ;ADDR OF WHERE TO RETURN TO
0102111' 000000  010177'
0102121' 000000  010206'
0102131' 001000  000305          33460   DVAR2:  PUSH    B               ;GOES ON STACK
```



```
                                 33480   IFE     LENGTH=2,<
0102141'                         33500   DVAR:
0102141' 001000  000257          33520   DVARS:  XRA     A
0102151' 001000  000266          33540           ORA     M               ;SEE IF ITS THE NULL STRING
0102161' 001000  000043          33560           INX     H
0102171' 001000  000136          33580           MOV     E,M
0102201' 001000  000043          33600           INX     H
0102211' 001000  000126          33620           MOV     D,M
0102221' 001000  000043          33640           INX     H>              ;(D,E)=POINTER AT THE VALUE
                                 33660   IFN     LENGTH=2,<
0102241' 000000  010166'         33680   DVAR:   ORI     128             ;FORCE DVAR TO CALL GRBVAR
                                 33700   DVARS:  PUSHM                   ;SAVE LENGTH
                                 33720           PUSHM                   ;SKIP NEXT TWO BYTES
                                 33740           POP     D               ;GET POINTER IN (D,E)
                                 33760           POP     B               ;POP OF STRING LENGTH
                                 33780           RP                      ;IF WASNT A STR, RETURN
0102251' 001000  000516          33800           MOV     A,C             ;GET LENGTH OF STRING
0102241' 001000  000240          33820           ORA     A>              ;SET CONDITION CODES
0102251' 001000  000115          33840           RZ                      ;NULL STRING, RETURN
0102261' 000000  001573'         33860           MOV     B,H             ;MOVE (H,L) TO (B,C)
0102271' 000000  010211'         33880           MOV     C,L
0102301' 000000  000052          33900           LHLD    FRETOP          ;GET POINTER TO TOP OF STRING FREE SPACE
0102311' 000000  010211'
0102321' 001000  000347          33920           COMPAR                  ;IS THIS STRINGS POINTER .LT. FRETOP
0102331' 001000  000140          33940           MOV     H,B             ;MOVE (B,C) BACK TO (H,L)
0102341' 001000  000151          33960           MOV     L,C
0102351' 001000  000330          33980           RC                      ;IF NOT, NO NEED TO MESS WITH IT FURTHUR
0102361' 001000  000341          34000           POP     H               ;GET RETURN ADDRESS OFF STACK
0102371' 001000  000343          34020           XTHL                    ;GET MAX SEEN SO FAR & SAVE RETURN ADDRESS
0102401' 001000  000347          34040           COMPAR                  ;LETS SEE
0102411' 001000  000343          34060           XTHL                    ;SAVE MAX SEEN & GET RETURN ADDRESS BACK
0102421' 001000  000345          34080           PUSH    H               ;SAVE RETURN ADDRESS BACK
0102431' 001000  000140          34100           MOV     H,B             ;MOVE (B,C) BACK TO (H,L)
0102441' 001000  000151          34120           MOV     L,C
0102451' 001000  000320          34140           RNC                     ;IF NOT, LETS LOOK AT NEXT VAR
0102461' 001000  000301          34160           POP     B               ;GET RETURN ADDR OFF STACK
0102471' 000000  000361          34180           POP     PSW             ;POP OFF MAX SEEN
0102501' 001000  000361          34200           POP     PSW             ;AND VARIABLE POINTER
0102511' 001000  000345          34220           PUSH    H               ;SAVE NEW VARIABLE POINTER
0102521' 001000  000325          34240           PUSH    D               ;AND NEW MAX POINTER
0102531' 001000  000305          34260           PUSH    B               ;SAVE RETURN ADDRESS BACK
0102541' 001000  000311          34280           RET                     ;AND RETURN
                                 34300   ;
                                 34320   ; HERE WHEN MADE ONE COMPLETE PASS THRU STRING VARS
                                 34340   ;
0102551' 001000  000321          34360   GWBPAS: POP     D               ;POP OFF MAX POINTER
0102561' 001000  000341          34380           POP     H               ;AND GET VARIABLE POINTER
0102571' 001000  000175          34400           MOV     A,L             ;SET LOW IN
0102601' 001000  000264          34420           ORA     H               ;SEE IF ZERO POINTER
0102611' 001000  000310          34440           RZ                      ;IF END OF COLLECTION,
                                 34460                                   ;THEN MAYBE RETURN TO GETSPA
0102621' 001000  000053          34480           DCX     H               ;CURRENTLY JUST PAST THE DESCRIPTOR
```

```
 6876  010262' 001000  000106          34500           MOV     B,M             ;[B]=HIGH BYTE OF DATA POINTER
 6877  010263' 001000  000053          34520           DCX     H
 6878  010264' 001000  000116          34540           MOV     C,M             ;[B,C]=POINTER AT STRING DATA
 6879  010265' 001000  000345          34550           PUSH    H               ;SAVE THIS LOCATION SO THE POINTER
 6880                                   34552           ;CAN BE UPDATED AFTER THE STRING IS
 6881                                   34554           ;MOVED
 6882  010266' 001000  000053          34560           DCX     H
 6883                                   34580   IFN     LENGTH-2,<
 6884                                   34600           DCX     H>
 6885  010267' 001000  000156          34620           MOV     L,M             ;[L]=STRING LENGTH
 6886  010270' 001000  000046          34640           MVI     H,0             ;[H,L] GET CHARACTER COUNT
 6887  010271' 000000  000000
 6888  010272' 001000  000011          34660           DAD     B               ;[H,L]=POINTER BEYOND STRING
 6889  010273' 001000  000120          34680           MOV     D,B
 6890  010274' 001000  000131          34700           MOV     E,C             ;[D,E]=ORIGINAL POINTER
 6891  010275' 001000  000053          34720           DCX     H               ;DON'T MOVE ONE BEYOND STRING
 6892  010276' 001000  000104          34740           MOV     B,H             ;GET TOP OF STRING IN [B,C]
 6893  010277' 001000  000115          34760           MOV     C,L
 6894  010300' 001000  000052          34780           LHLD    FRETOP          ;GET TOP OF FREE SPACE
 6895  010301' 000000  010273'
 6896  010302' 000000  010227'
 6897  010305' 001000  000315          34800           CALL    BLTUC           ;MOVE STRING
 6898  010304' 000000  002010'
 6899  010305' 000000  010301'
 6900  010306' 001000  000341          34820           POP     H               ;GET BACK POINTER TO DESC.
 6901  010307' 001000  000161          34840           MOV     M,C             ;SAVE FIXED ADDR
 6902  010310' 001000  000043          34860           INX     H               ;MOVE POINTER
 6903  010311' 001000  000160          34880           MOV     M,B             ;HIGH PART
 6904  010312' 001000  000151          34900           MOV     L,C             ;[H,L]=NEW POINTER
 6905  010313' 001000  000140          34920           MOV     H,B             ;FIX UP FRETOP
 6906  010314' 001000  000053          34940           DCX     H               ;AND TRY TO FIND HIGH AGAIN
 6907  010315' 001000  000303          34960           JMP     FNDVAR
 6908  010316' 000000  010045'
 6909  010317' 000000  010304'
 6910
 6911                                   35000   ;
 6912                                   35020   ; THE FOLLOWING ROUTINE CONCATENATES TWO STRINGS
 6913                                   35040   ; THE FACLO CONTAINS THE FIRST ONE AT THIS POINT,
 6914                                   35060   ; [H,L] POINTS BEYOND THE + SIGN AFTER IT
 6915                                   35080   ;
 6916  010320' 001000  000305          35100   CAT:    PUSH    B               ;PUT OLD PRECEDENCE BACK ON
 6917  010321' 001000  000345          35120           PUSH    H               ;SAVE TEXT POINTER
 6918  010322' 001000  000052          35140           LHLD    FACLO           ;GET POINTER TO STRING DESC.
 6919  010325' 000000  001637'
 6920  010324' 000000  010316'
 6921  010325' 001000  000343          35160           XTHL                    ;SAVE ON STACK & GET TEXT POINTER BACK
 6922  010326' 001000  000315          35180           CALL    EVAL            ;EVALUATE REST OF FORMULA
 6923  010327' 000000  006661'
 6924  010330' 000000  010323'
 6925  010331' 001000  000343          35200           XTHL                    ;SAVE TEXT POINTER, GET BACK DESC.
 6926  010332' 001000  000315          35220           CALL    CHKSTR
 6927  010335' 000000  006630*
 6928  010334' 000000  010327'
```



```
 6929  010335' 001000  000176          35240           MOV     A,M
 6930  010336' 001000  000345          35260           PUSH    H               ;SAVE DESC. POINTER,
 6931  010337' 001000  000052          35280           LHLD    FACLO           ;GET POINTER TO 2ND DESC.
 6932  010340' 000000  001637'
 6933  010341' 000000  010335'
 6934  010342' 001000  000345          35300           PUSH    H               ;SAVE IT
 6935  010343' 001000  000206          35320           ADD     M               ;ADD TWO LENGTHS TOGETHER
 6936  010344' 001000  000636          35340           MVI     E,ERRLS         ;SEE IF RESULT .LT. 256
 6937  010345' 000000  000017
 6938  010346' 001000  000332          35360           JC      ERROR           ;ERROR "LONG STRING"
 6939  010347' 000000  002102'
 6940  010350' 000000  010340'
 6941  010351' 001000  000315          35380           CALL    STRINI          ;GET INITIAL STRING
 6942  010352' 000000  007624'
 6943  010353' 000000  010347'
 6944  010354' 001000  000321          35400           POP     D               ;GET 2ND DESC.
 6945  010355' 001000  000315          35420           CALL    FRETMP
 6946  010356' 000000  010440'
 6947  010357' 000000  010352'
 6948  010360' 001000  000343          35440           XTHL                    ;SAVE POINTER TO IT
 6949  010361' 001000  000315          35460           CALL    FRETM2          ;FREE UP 1ST TEMP
 6950  010362' 000000  010437'
 6951  010363' 000000  010356'
 6952  010364' 001000  000345          35480           PUSH    H               ;SAVE DESC. POINTER (FIRST)
 6953  010365' 001000  000052          35500           LHLD    DSCTMP+2        ;GET POINTER TO FIRST
 6954  010366' 000000  001637'
 6955  010367' 000000  010362'
 6956  010370' 001000  000353          35520           XCHG                    ;IN [D,E]
 6957  010371' 001000  000315          35540           CALL    MOVINS          ;MOVE IN THE FIRST STRING
 6958  010372' 000000  010407'
 6959  010373' 000000  010366'
 6960  010374' 001000  000315          35560           CALL    MOVINS          ;AND THE SECOND
 6961  010375' 000000  010407'
 6962  010376' 000000  010372'
 6963  010377' 001000  000041          35580           LXI     H,TSTOP         ;CAT REENTERS FORMULA EVALUATION AT TSTOP
 6964  010400' 000000  065357'
 6965  010401' 000000  010375'
 6966  010402' 001000  000343          35600           XTHL
 6967  010403' 001000  000345          35620           PUSH    H               ;TEXT POINTER OFF FIRST
 6968  010404' 001000  000303          35640           JMP     PUTNEW          ;THEN RETURN ADDRESS OF TSTOP
 6969  010405' 000000  007705'
 6970  010406' 000000  010400'
 6971
 6972
 6973  010407' 001000  000341          35700   MOVINS: POP     H               ;GET RETURN ADDR
 6974  010410' 001000  000343          35720           XTHL                    ;PUT BACK, BUT GET DESC.
 6975                                   35740   IFE     LENGTH-2,<
 6976  010411' 001000  000176          35760           MOV     A,M             ;[A]=STRING LENGTH
 6977  010412' 001000  000043          35780           INX     H
 6978  010413' 001000  000116          35800           MOV     C,M             ;[B,C]=POINTER AT STRING DATA
 6979  010414' 001000  000043          35820           INX     H
 6980  010415' 001000  000106          35840           MOV     B,M
 6981  010416' 001000  000137          35860           MOV     L,A>            ;[L]=STRING LENGTH
```

```
                                        35880   IFN     LENGTH=2,<
                                        35900           PUSHM                   ;GET LENGTH ON STACK
                                        35920           PUSHM                   ;AND POINTER
                                        35940           POP     B               ;TEXT POINTER HERE
                                        35960           POP     H>              ;CHARACTER COUNT HERE
6987  010417' 001000  000054           35980   MOVSTR: INR     L
6988  010420' 001000  000055           36000   MOVLP:  DCR     L               ;SET CC'S
6989  010421' 001000  000310           36020           RZ                      ;0, NO BYTE TO MOVE
6990  010422' 001000  000012           36040           LDAX    B               ;GET CHAR
6991  010423' 001000  000002           36060           STAX    D               ;SAVE IT
6992  010424' 001000  000003           36080           INX     B               ;MOVE POINTERS
6993  010425' 001000  000023           36100           INX     D
6994  010426' 001000  000303           36120           JMP     MOVLP           ;KEEP DOING IT
6995  010427' 000000  010420'
6996  010430' 000000  010405'
6997                                    36140   ;
6998                                    36160   ; FRETMP IS PASSED A POINTER TO A STRING DESCRIPTOR IN (D,E)
6999                                    36180   ; THIS VALUE IS RETURNED IN (H,L), ALL THE OTHER REGISTERS ARE MODIFIED,
7000                                    36200   ; A CHECK TO IS MADE TO SEE IF THE STRING DESCRIPTOR (D,E) POINTS
7001                                    36220   ; TO IS THE LAST TEMPORARY DESCRIPTOR ALLOCATED BY PUTNEW.
7002                                    36240   ; IF SO, THE TEMPORARY IS FREED UP BY THE UPDATING OF TEMPPT.
7003                                    36260   ; IF A TEMPORARY IS FREED UP, A FURTHER CHECK IS MADE TO SEE IF THE
7004                                    36280   ; STRING DATA THAT THAT STRING TEMPORARY POINTED TO IS THE
7005                                    36300   ; THE LOWEST PART OF STRING SPACE IN USE.
7006                                    36320   ; IF SO, FRETMP IS UPDATED TO REFLECT THE FACT THAT THAT SPACE IS NO
7007                                    36340   ; LONGER IS USE. THIS CAUSES DIFFICULTY FOR ASSIGNMENT ("LET") BECAUSE
7008                                    36360   ; THOUGH A TEMPORARY IS BEING FREED UP, NAMELY THE VALUE TO THE RIGHT
7009                                    36380   ; OF THE EQUAL SIGN IN THE "LET", THE ACTUAL DATA
7010                                    36400   ; IS STILL ACTIVE DATA SINCE A VARIABLE IS BEING SET UP TO POINT
7011                                    36420   ; AT IT. "LET" FOOLS FRETMP BY SETTING THE LENGTH OF THE
7012                                    36440   ; TEMPORARY TO ZERO TEMPORARILY.
7013                                    36460   ;
7014  010431' 001000  000315           36480   FRESTR: CALL    CHKSTR          ;MAKE SURE ITS A STRING
7015  010432' 000000  010333*
7016  010433' 000000  010427'
7017  010434' 001000  000052           36500   FREFAC: LHLD    FACLO
7018  010435' 000000  001637'
7019  010436' 000000  010432'
7020  010437' 001000  000353           36520   FRETMP: XCHG                    ;FREE UP THE TEMP IN THE FACLO
7021  010440' 001000  000052           36540   FRETM1: LHLD    TEMPPT          ;GET TEMP POINTER
7022  010441' 000000  001547'
7023  010442' 000000  010435'
7024  010443' 001000  000053           36560           DCX     H               ;LOOK AT WHAT IS IN THE LAST TEMP
7025  010444' 001000  000106           36580           MOV     B,M             ;(B,C)=POINTER AT STRING
7026  010445' 001000  000053           36600           DCX     H               ;DECREMENT TEMPPT BY STRSIZ
7027  010446' 001000  000116           36620           MOV     C,M
7028  010447' 001000  000053           36640           DCX     H
7029                                    36660   IFN     LENGTH=2,<
7030                                    36680           DCX     H>
7031  010450' 001000  000347           36700           COMPAR                  ;SEE IF (D,E) POINT AT THE LAST
7032  010451' 001000  000353           36720           XCHG                    ;RETURN WITH (H,L)
7033                                    36740                                   ;POINTING AT CURRENT DESCRIPTOR
7034  010452' 001000  000300           36760           RNZ                     ;RETURN NOW IF NOW FREEING SOME
```



```
7035  010453' 001000  000042           36780           SHLD    TEMPPT          ;UPDATE THE TEMP POINTER SINCE
7036  010454' 000000  001547'
7037  010455' 000000  010441'
7038                                    36800                                   ;ITS BEEN DECREMENTED BY 4
7039  010456' 001000  000325           36820           PUSH    D               ;SAVE (D,E) TO RETURN IN (H,L)
7040  010457' 001000  000120           36840           MOV     D,B             ;(D,E)=POINTER AT STRING
7041  010460' 001000  000131           36860           MOV     E,C
7042  010461' 001000  000033           36880           DCX     D               ;SUBTRACT ONE
7043  010462' 001000  000116           36900           MOV     C,M             ;(C)=LENGTH OF THE STRING FREED UP
7044  010463' 001000  000052           36920           LHLD    FRETOP          ;SEE IF ITS THE FIRST
7045  010464' 000000  001575'
7046  010465' 000000  010454'
7047                                    36940                                   ;ONE IN STRING SPACE
7048  010466' 001000  000347           36960           COMPAR
7049  010467' 001000  000302           36980           JNZ     NOTLST          ;NO SO DON'T ADD
7050  010470' 000000  010471'
7051  010471' 000000  010464'
7052  010472' 001000  000107           37000           MOV     B,A             ;MAKE (B)=0
7053  010473' 001000  000011           37020           DAD     B               ;ADD
7054  010474' 001000  000042           37040           SHLD    FRETOP          ;AND UPDATE FRETOP
7055  010475' 000000  001573'
7056  010476' 000000  010470'
7057  010477' 001000  000341           37060   NOTLST: POP     H               ;GET POINTER AT CURRENT DESCRIPTOR
7058  010500' 001000  000311           37080           RET
7059                                    37100   ;
7060                                    37120   ; THE FUNCTION LEN($) RETURNS THE LENGTH OF THE
7061                                    37140   ; STRING PASSED AS AN ARGUMENT
7062                                    37160   ;
7063  010501' 001000  000001           37180   LEN:    LXI     B,SNGFLT        ;CALL SNGFLT WHEN DONE
7064  010502' 000000  007400'
7065  010503' 000000  010475'
7066  010504' 001000  000325           37200           PUSH    B               ;LIKE SO
7067  010505' 001000  000315           37220   LEN1:   CALL    FRESTR          ;FREE UP TEMP POINTED TO BY FACLO
7068  010506' 000000  010431'
7069  010507' 000000  010506'
7070  010510' 001000  000257           37240           XRA     A               ;FORCE NUMERIC FLAG
7071  010511' 001000  000127           37260           MOV     D,A             ;SET HIGH OF (D,E) TO ZERO FOR VAL
7072                                    37280   IFN     LENGTH=2,<
7073                                    37300           STA     VALTYP>
7074  010512' 001000  000176           37320           MOV     A,M             ;SET CONDITION CODES ON LENGTH
7075  010513' 001000  000267           37340           ORA     A               ;RETURN
7076  010514' 001000  000311           37360           RET
7077                                    37380   ;
7078                                    37400   ; THE FOLLOWING IS THE ASC($) FUNCTION. IT RETURNS AN INTEGER
7079                                    37420   ; WHICH IS THE DECIMAL ASCII EQUIVALENT
7080                                    37440   ;
7081  010515' 001000  000315           37460   ASC:    CALL    LEN1            ;SET UP ORIGINAL STR
7082  010516' 000000  010505'
7083  010517' 000000  010586'
7084  010520' 001000  000312           37480           JZ      FCERR           ;NULL STR, BAD ARG.
7085  010521' 000000  010776'
7086  010522' 000000  010516'
7087  010523' 001000  000043           37500           INX     H               ;BUMP POINTER
```

```
                                        37520   IFN     LENGTH-2,<
                                        37540           INX     H>              ;BUMP POINTER
7090  010524' 001000  000567            37560           PUSHM                   ;GET ADDRESS
7091  010525' 001000  000341            37580           POP     H               ;GET ADDR IN (H,L)
7092  010526' 001000  000176            37600           MOV     A,M             ;GET FIRST CHAR
7093  010527' 001000  000303            37620           JMP     SNGFLT          ;SNGFLT IT
7094  010530' 000000  007400'
7095  010531' 000000  010521'
7096                                    37640   ;
7097                                    37660   ; CHR$(#) CREATES A STRING WHICH CONTAINS AS ITS ONLY
7098                                    37680   ; CHARACTER THE ASCII EQUIVALENT OF THE INTEGER ARG (#)
7099                                    37700   ; WHICH MUST BE .LT. 255.
7100                                    37720   ;
7101  010532' 001000  000076            37740   CHR$:   MVI     A,1             ;ONE CHARACTER STR
7102  010533' 000002  000001
7103  010534' 001000  000315            37760           CALL    STRIN1          ;GET STRING IN DSCTMP
7104  010535' 000000  007624'
7105  010536' 000000  010530'
7106  010537' 001000  000315            37780           CALL    CONINT          ;GET INTEGER IN RANGE
7107  010540' 000000  011025'
7108  010541' 000000  010535'
7109  010542' 001000  000052            37800           LHLD    DSCTMP+2        ;GET ADDR OF STR
7110  010543' 000000  001572'
7111  010544' 000000  010540'
7112  010545' 001000  000163            37820           MOV     M,E             ;SAVE ASCII BYTE
7113  010546' 001000  000301            37840   FINBCK: POP     B               ;RETURN TO HIGHER LEVEL B
7114                                    37860           ;SKIP THE CHKNUM CAL.,
7115  010547' 001000  000303            37880           JMP     PUTNEW          ;GO CALL PUTNEW
7116  010550' 000000  007705'
7117  010551' 000000  010543'
7118                                    37900   ;
7119                                    37920   ; THE FOLLOWING IS THE LEFT$(S,#) FUNCTION.
7120                                    37940   ; IT TAKES THE LEFTMOST # CHARS OF THE STR,
7121                                    37960   ; IF # IS .GT. THAN THE LEN OF THE STR, IT RETURNS THE WHOLE STR,
7122                                    37980   ;
7123  010552' 001000  000315            38000   LEFT$:  CALL    PREAM           ;TEST THE PARAMETERS
7124  010553' 000000  010764'
7125  010554' 000000  010550'
7126  010555' 001000  000257            38020           XRA     A               ;.LEFT NEVER CHANGES STRING POINTER
7127  010556' 001000  000343            38040   LEFT3:  XTHL                    ;SAVE TEXT POINTER
7128  010557' 001000  000117            38060           MOV     C,A             ;OFFSET NOW IN (C)
7129  010560' 001000  000345            38080   LEFT2:  PUSH    H               ;SAVE DESC. FOR  FRETMP
7130  010561' 001000  000176            38100           MOV     A,M             ;GET STRING LENGTH
7131  010562' 001000  000270            38120           CMP     B               ;ENTIRE STRING WANTED?
7132  010563' 001000  000332            38140           JC      ALLSTR          ;IF #CHARS ASKED FOR.GE.LENGTH,YES
7133  010564' 000000  010570'
7134  010565' 000000  010553'
7135  010566' 001000  000170            38160           MOV     A,B             ;GET TRUNCATED LENGTH OF STRING
7136  010567' 001000  000021            38180           DW      "010000,"021    ;SKIP OVER MVI USING "LXI D,"
7137  010570' 001000  000016            38200   ALLSTR: MVI     C,0             ;MAKE OFFSET ZERO
7138  010571' 000000  000000
7139  010572' 001000  000305            38220           PUSH    B               ;SAVE OFFSET ON STACK
7140  010573' 001000  000315            38240           CALL    GETSPA          ;GET SPACE FOR NEW STRING
```



```
7141  010574' 000000  007777'
7142  010575' 000000  010564'
7143  010576' 001000  000301            38260           POP     B               ;GET BACK OFFSET
7144  010577' 001000  000341            38280           POP     H               ;GET BACK DESC POINTER,
7145  010600' 001000  000345            38300           PUSH    H               ;BUT KEEP ON STACK
7146  010601' 001000  000043            38320           INX     H               ;MOVE TO STRING POINTER FIELD
7147                                    38340   IFN     LENGTH-2,<
7148                                    38360           INX     H>
7149  010602' 001000  000106            38380           MOV     B,M             ;GET POINTER LOW
7150  010603' 001000  000043            38400           INX     H               ;
7151  010604' 001000  000146            38420           MOV     H,M             ;POINTER HIGH
7152  010605' 001000  000150            38440           MOV     L,B             ;GET LOW IN  L
7153  010606' 001000  000006            38460           MVI     B,0             ;GET READY TO ADD OFFSET TO POINTER
7154  010607' 000000  000000
7155  010610' 001000  000011            38480           DAD     B               ;ADD  IT
7156  010611' 001000  000104            38500           MOV     B,H             ;GET OFFSET POINTER IN (B,C)
7157  010612' 001000  000115            38520           MOV     C,L
7158  010613' 001000  000315            38540           CALL    STRAD2          ;SAVE INFO IN DSCTMP
7159  010614' 000000  007627'
7160  010615' 000000  010574'
7161  010616' 001000  000157            38560           MOV     L,A             ;GET#  OF CHARS TO  MOVE IN L
7162  010617' 001000  000315            38580           CALL    MOVSTR          ;MOVE THEM IN
7163  010620' 000000  010417'
7164  010621' 000000  010614'
7165  010622' 001000  000321            38600           POP     D               ;GET BACK DESC. POINTER
7166  010623' 001000  000315            38620           CALL    FRETMP          ;FREE IT UP,
7167  010624' 000000  010440'
7168  010625' 000000  010620'
7169  010626' 001000  000303            38640           JMP     PUTNEW          ;PUT TEMP IN TEMP LIST
7170  010627' 000000  007765'
7171  010630' 000000  010624'
7172
7173  010631' 001000  000315            38660   RIGHT$: CALL    PREAM           ;CHECK ARG
7174  010632' 000000  010764'
7175  010635' 000000  010627'
7176  010634' 001000  000321            38700           POP     D               ;GET DESC. POINTER
7177  010635' 001000  000325            38720           PUSH    D               ;SAVE BACK FOR LEFT
7178  010636' 001000  000032            38740           LDAX    D               ;GET PRESENT LEN OF STR
7179  010637' 001000  000220            38760           SUB     B               ;SUBTRACT 2ND PARM
7180  010640' 001000  000303            38780           JMP     LEFT3           ;CONTINUE WITH LEFT CODE
7181  010641' 000000  010556'
7182  010642' 000000  010632'
7183                                    38800   ;
7184                                    38820   ; MID ($,#) RETURNS STR WITH CHARS FROM # POSITION
7185                                    38840   ; ONWARD, IF # IS GT LEN($) THEN RETURN NULL STRING,
7186                                    38860   ; MID ($,#,#) RETURNS STR WITH CHARS FROM # POSITION
7187                                    38880   ; FOR #2 CHARS, IF #2 GOES PAST END OF STRING, RETURN
7188                                    38900   ; AS MUCH AS POSSIBLE.
7189                                    38920   ;
7190  010643' 001000  000353            38940   MID$:   XCHG                    ;PUT THE TEXT POINTER IN (H,L)
7191  010644' 001000  000176            38960           MOV     A,M             ;GET THE FIRST CHARACTER
7192  010645' 001000  000315            38980           CALL    PREAM2          ;GET OFFSET OFF STACK AND MAKE
7193  010646' 000000  010767'
```

```
                                                        ;SUBR DOES NOT # 0.
                                                        ;PUT OFFSET ON TO THE STACK
                                                        ;IF TWO ARG GUY, TRUNCATE.
                                                        ;IF ONE ARGUMENT THIS IS CORRECT
                                                        ;COMMAT MUST DELINEATE 3RD ARG.
                                                        ;GET ARGUMENT  IN  (E)
                                                        ;(E) SAYS USE ALL CHARS
                                                        ;MUST BE FOLLOWED BY )
                                                        ;GET OFFSET BACK IN A
                                                        ;SAVE TEXT POINTER, GET DESC.
                                                        ;WHERE TO RETURN TO.
                                                        ;DOES ON STACK
                                                        ;SUB ONE FROM OFFSET
                                                        ;POINTER PAST END OF STR?
                                                        ;ASSUME NULL LENGTH STR
                                                        ;YES, JUST USE NULL STR
                                                        ;SAVE OFFSET OF CHARACTER POINTER
                                                        ;GET PRESENT LEN OF STR
                                                        ;SUBTRACT INDEX (2ND ARG)
                                                        ;IS IT TRUNCATION
                                                        ;SET CALCED LENGTH IN B
                                                        ;IF NOT USE PARTIAL STR
                                                        ;USE TRUNCATED LENGTH
                                                        ;RETURN TO LEFT2
IFN LENGTH,<
;
; THE FOLLOWING FUNCTIONS ALLOW THE
; USER FULL ACCESS TO THE ALTAIR I/O PORTS
; INP(CHANNEL#) RETURNS AN INTEGER WHICH IS THE STATUS
; OF THE CHANNEL. OUT CHANNEL#,VALUE PUTS OUT THE INTEGER
; VALUE ON CHANNEL #. IT IS A STATEMENT, NOT A FUNCTION.
;
FNINP: CALL    CONINT                   ;GET INTEGER CHANNEL #

       STA     INPWRD+1                 ;GEN INP INSTR

INPWRD: IN     0                        ;THE INP INSTR

       RET>
```



```
       JMP     SNGFLT                   ;SNGFLT RESULT

FNOUT: CALL    SETIO                    ;GET READY

       STA     OUTWRD+1                 ;DO IT

OUTWRD: OUT    0

       RET                              ;AND THATS ALL
;
; THE WAIT CHANNEL#,MASK,MASK2 WAITS UNTIL THE STATUS
; RETURNED BY CHANNEL# IS NON ZERO WHEN XORED WITH MASK2
; AND THEN ANDED WITH MASK, IF MASK2 IS NOT PRESENT IT IS ASSUMED
; TO BE ZERO.
;
FNWAIT: CALL   SETIO                    ;SET UP FOR WAIT

       PUSH    PSW                      ;SAVE THE MASK
       MVI     E,0                      ;DEFAULT MASK2 TO ZERO

       DCX     H
       CHRGET                           ;SEE IF THE STATEMENT ENDED
       JZ      NOTTHR                   ;IF NO THIRD ARGUMENT SKIP THIS

       SYNCHK  44                       ;MAKE SURE THERE IS A ","

       CALL    GETBYT

       POP     B                        ;REGET THE "AND" MASK

STAINP: IN     0                        ;THE INPUT INSTR

       XRA     E                        ;XOR WITH MASK2
       ANA     B                        ;AND WITH MASK

       JZ      STAINP                   ;LOOP UNTIL RESULT IS NON-ZERO

                                        ;NOTE: THIS LOOP CANNOT BE CONTROL-C'ED
                                        ;UNLESS THE WAIT IS BEING DONE ON CHANNEL
                                        ;ZERO. HOWEVER A RESTART AT 0 IS OK.
       RET>

IFN    STRING,<

;USED BY MIGHTS AND LEFTS FOR PARAMETER CHECKING AND SETUP

PREAM: XCHG                             ;PUT THE TEXT POINTER IN (H,L)
       SYNCHK  ")"                      ;PARAM LIST SHOULD END

;USED BY MIDS FOR PARAMETER CHECKING AND SETUP

PREAM2: POP    B                        ;GET RETURN ADDR OFF STACK
       POP     D                        ;GET LENGTH OF ARG OFF STACK
       PUSH    B                        ;SAVE RETURN ADDR BACK ON
```

```
 7300  011771' 001000  000103          40440          MOV     B,E             ;SAVE INIT LENGTH
 7301  011772' 001000  000004          40460          INR     B
 7302  011773' 001000  000005          40480          DCR     B               ;SEE IF EQUAL TO ZERO
 7303                                   40500  IFE     LENGTH-2,<
 7304  011775' 001000  000340          40520          RNZ
 7305  011776'                        40536  ILLFUN:
 7306  011776' 001000  000036          40540  FCERR1: MVI     E,ERRFC
 7307  012000' 000000  000005
 7308  012001' 001000  000323          40560          JMP     ERROR>
 7309  012002' 000000  000104'
 7310  012004' 000000  010761'
 7311                                   40580  IFN     LENGTH-2,<
 7312                                   40600          JZ      FCERR           ;IT MUST NOT BE 0
 7313                                   40620          RET>
 7314                                   40640  IFN     LENGTH,<
 7315  012005' 001000  000315          40660  SETIO:  CALL    GETBYT          ;GET INTEGER CHANNEL NUMBER IN (A)
 7316  012006' 000000  011E20'
 7317  012007' 000000  011001'
 7318  012008' 001000  000066          40680          STA     STAINP+1        ;SETUP "WAIT"
 7319  012009' 000000  012755'
 7320  012010' 000000  011034'
 7321  012011' 001000  000062          40700          STA     OUTWRD+1        ;SETUP "OUT"
 7322  012012' 000000  010731'
 7323  012013' 000000  011007'
 7324  012014' 001000  000317          40720          SYNCHK  44              ;MAKE SURE THERE IS A COMMA
 7325  012015' 000000  000054
 7326  012016' 001000  000006          40740          XWD     "D1000,6>       ;"MVI B," AROUND THE CHRGET
 7327                                   40760  IFN     STRING!LENGTH,<
 7328  012017' 001000  000327          40780  GTBYTC: CHRGET
 7329                                   40800  IFE     LENGTH-2,<
 7330  012020' 001000  000315          40820  GETBYT: CALL    FRMEVL
 7331  012021' 000000  065336'
 7332  012022' 000000  011016'
 7333  012023' 001000  000345          40840  CONINT: PUSH    H
 7334  012024' 001000  000315          40860          CALL    FRCINT
 7335  012025' 000000  066441'
 7336  012026' 000000  011021'
 7337  012027' 001000  000353          40880          XCHG
 7338  012030' 001000  000341          40900          POP     H>
 7339                                   40920  IFN     LENGTH-2,<
 7340                                   40940  GETBYT: CALL    FRMNUM          ;READ FORMULA INTO THE FAC.
 7341                                   40960  CONINT: CALL    POSINT          ;CONVERT THE FAC TO A SINGLE BYTE INTEGER
 7342  012031' 001000  000172          40980          MOV     A,D             ;SHOULD BE .LT. 255
 7343  012032' 001000  000267          41000          ORA     A               ;SET CC'S
 7344  012033' 001000  000302          41020          JNZ     FCERR           ;WASN'T ERROR
 7345  012034' 000000  010776'
 7346  012035' 000000  011025'
 7347  012036' 001000  000053          41040          DCX     H               ;ACTUALLY FUNCTIONS CAN GET HERE
 7348                                   41060                                  ;WITH BAD (H,L) BUT NOT SERIOUS
 7349                                   41080                                  ;SET CONDITION CODES ON TERMINATOR
 7350  012037' 001000  000327          41100  CHRGET
 7351  012040' 001000  000173          41120          MOV     A,E             ;RETURN THE RESULT IN (A) AND (E)
 7352  012041' 001000  000311          41140          RET>
```



```
 7353
 7354                                   41160  IFN     STRING,<
 7355                                   41200  ;
 7356                                   41202  ; THE VAL FUNCTION TAKES A STRING AND TURN IT INTO
 7357                                   41204  ; A NUMBER BY INTERPRETING THE ASCII DIGITS, ETC.,
 7358                                   41206  ; EXCEPT FOR THE PROBLEM THAT A TERMINATOR MUST BE SUPPLIED
 7359                                   41208  ; BY REPLACING THE CHARACTER BEYOND THE STRING, VAL
 7360                                   41210  ; IS MERELY A CALL TO FLOATING INPUT (FIN),
 7361                                   41212  ;
 7362  012042' 001000  000315          41220  VAL:    CALL    LEN1            ;DO SETUP, SET RESULT=REAL
 7363  012043' 000000  010505'
 7364  012044' 000000  011034'
 7365  012045' 001000  000312          41240          JZ      ZERO            ;RETURN ZERO IF NULL
 7366  012046' 000000  000000*
 7367  012047' 000000  011045'
 7368  012050' 001000  000137          41260          MOV     E,A             ;GET LENGTH OF STR
 7369                                   41270  IFN     LENGTH-2,<
 7370                                   41280          INX     H>              ;THIS IS ALL A KLUDGE
 7371  012051' 001000  000043          41300          INX     H               ;TO HANDLE THE FACT THE IF
 7372  012052' 001000  000367          41320          PUSHM                   ;TWO STRINGS "1" AND "2"
 7373  012053' 001000  000140          41340          MOV     H,B             ;ARE STORED NEXT TO EACH OTHER
 7374  012054' 001000  000151          41360          MOV     L,C             ;AND FIN IS CALLED POINTING TO
 7375  012055' 001000  000031          41380          DAD     D               ;THE FIRST TWELVE WILL BE RETURNED
 7376  012056' 001000  000126          41400          MOV     B,M             ;THE IDEA IS TO STORE 0 IN THE
 7377  012057' 001000  000162          41420          MOV     M,D             ;STRING BEYOND THE ONE VAL
 7378  012060' 001000  000343          41440          XTHL                    ;IS BEING CALLED ON
 7379  012061' 001000  000305          41460          PUSH    B               ;THE FIRST CHARACTER OF THE NEXT STRING
 7380  012062' 001000  000176          41480          MOV     A,M             ;GET FIRST CHARACTER OF ARGUMENT
 7381  012063' 001000  000315          41500          CALL    FIN             ;TURN IT INTO A NUMBER IN THE FAC
 7382  012064' 000000  006103*
 7383  012065' 000000  011046'
 7384  012066' 001000  000301          41520          POP     B               ;GET THE MODIFIED CHARACTER OF THE NEXT
 7385                                   41540                                  ;STRING INTO (B)
 7386  012067' 001000  000341          41560          POP     H               ;GET THE POINTER TO THE MODIFIED CHARACTER
 7387  012070' 001000  000160          41580          MOV     M,B             ;RESTORE THE CHARACTER
 7388                                   41600                                  ;IF STRING IS HIGHEST IN STRING SPACE
 7389                                   41620                                  ;WE ARE MODIFYING (MEMSIZ) AND
 7390                                   41640                                  ;THIS IS WHY (MEMSIZ) CAN'T BE USED TO STORE
 7391                                   41660                                  ;STRING DATA BECAUSE WHAT IF THE
 7392                                   41680                                  ;USER TOOK VAL OFF THAT HIGH STRING
 7393  012071' 001000  000311          41700          RET>
 7394
 7395                                   41740  PAGE
```

```
                                41760   SUBTTL  FANCY LIST,DELETE,EDIT,LLIST
                                41780   IFE     LENGTH-2,<
                                41800   IFN     LPTSW,<
 7400 011072' 001000  000301    41820   LLIST:  MVI     A,1             ;SET NON ZERO VALUE
 7401 011074' 001000  000315    41840           STA     PRTFLG>         ;SAVE IN I/O FLAG
 7402 011075' 001000  002333'   41860   LIST:   POP     B               ;GET RID OF NEXT RETURN ADDR
 7403 011076' 000000  011664'   41880           CALL    SCNLIN          ;SCAN LINE RANGE
 7404 011079' 000000
 7405 011076' 001000  000305    41900           PUSH    B               ;SAVE POINTER TO 1ST LINE
 7406 011077' 001000  000341    41920   LIST4:  POP     H               ;GET POINTER TO LINE
 7407 011100' 001000  000321    41940           POP     D               ;GET MAX LINE # OFF STACK
 7408 011101' 001000  000367    41960           PUSHM                   ;PUSH LINK
 7409 011102' 001000  000170    41980           MOV     A,B             ;SEE IF END OF CHAIN
 7410 011103' 001000  000261    42000           ORA     C
 7411 011104' 001000  000301    42020           POP     B               ;GET LINK OFF STACK FOR ISCNTC
 7412 011105' 001000  000512    42040           JZ      READY           ;LAST LINE, STOP,
 7413 011106' 000000  002145'
 7414 011107' 000000  011674'
 7415                           42060   IFN     LISTEN,<
 7416 011110' 001000  000315    42080           CALL    ISCNTC>         ;CHECK FOR CONTROL-C
 7417 011111' 000000  003460'
 7418 011112' 000000  011106'
 7419 011113' 001000  000305    42100           PUSH    B               ;SAVE LINK BACK ON
 7420 011114' 001000  000367    42120           PUSHM                   ;PUSH LINE #
 7421 011115' 001000  000343    42140           XTHL                    ;GET LINE # INTO [H,L]
 7422 011116' 001000  000353    42160           XCHG                    ;GET MAX LINE IN [H,L]
 7423 011117' 001000  000247    42180           COMPAR                  ;PAST LAST LINE IN RANGE?
 7424 011120' 001000  000301    42200           POP     B               ;TEXT POINTER TO [B,C]
 7425 011121' 001000  000332    42220           JC      STPROT          ;IF PAST, THEN DONE LISTING,
 7426 011122' 000000  002144'
 7427 011123' 000000  011111'
 7428 011124' 001000  000343    42280           XTHL                    ;SAVE MAX ON BOTTOM OF STACK
 7429 011125' 001000  000345    42300           PUSH    H               ;SAVE LINK ON TOP
 7430 011126' 001000  000305    42320           PUSH    B               ;SAVE TEXT POINTER BACK
 7431 011127' 001000  000353    42340           XCHG                    ;SET LINE # IN [H,L]
 7432 011130' 001000  000315    42360           CALL    CRDO            ;DO CRLF TO START OUT
 7433 011131' 000000  004523'
 7434 011134' 000000  011126'
 7435                           42380                                   ;AND WE WANT [H,L] ON THE STACK
 7436 011133' 001000  000315    42400           CALL    LINPRT          ;PRINT AS INT WITHOUT LEADING SPACE
 7437 011134' 000000  003561*
 7438 011135' 000000  011131'
 7439 011136' 001000  000076    42420           MVI     A," "
 7440 011137' 000000  000040
 7441 011140' 001000  000341    42440           POP     H
 7442 011141' 001000  000537    42460           OUTCHR                  ;PRINT A SPACE AFTER THE LINE #
 7443 011142' 001000  000315    42480           CALL    BUFLIN          ;UNPACK THE LINE INTO BUF
 7444 011143' 000000  011163'
 7445 011144' 000000  011134'
 7446 011145' 001000  000041    42500           LXI     H,BUF-1         ;POINT AT THE START OF THE UNPACKED CHARACTERS
 7447 011146' 000000  011143'
 7448 011147' 000000  011143'
```



```
 7449 011150' 001000  000006    42520           MVI     B,0             ;STOP ON ZERO ONLY
 7450 011151' 000000  000000
 7451 011152' 001000  000315    42540           CALL    STRLT3          ;INITIALIZE THE LINE STRING
 7452 011155' 000000  007664'
 7453 011156' 000000  011146'
 7454 011159' 001000  000315    42560           CALL    STRPRT          ;PRINT OUT THE CHARACTERS
 7455 011156' 000000  007746'
 7456 011157' 000000  011153'
 7457 011160' 001000  000303    42580           JMP     LIST4           ;PRINT ANOTHER LINE
 7458 011161' 000000  011077'
 7459 011166' 000000  011156'
 7460 011163' 001000  000001    42600   BUFLIN: LXI     B,BUF-1
 7461 011164' 000000  001430'
 7462 011165' 000000  011161'
 7463 011166' 001000  000020    42620           XWD     "D1000,^O26     ;"MVI   D," AROUND THE NEXT BYTE
 7464 011167' 001000  000341    42640   PRIT4:  POP     H               ;RESTORE POINTER TO START OF TEXT
 7465 011170' 001000  000176    42660   PLOOP:  MOV     A,M             ;GET A CHARACTER FROM LINE,
 7466 011171' 001000  000003    42680           INX     B               ;ADVANCE STUFF COUNT
 7467 011172' 001000  000267    42700           ORA     A               ;IS IT A RESERVED WORD
 7468 011173' 001000  000043    42720           INX     H               ;INCREMENT POINTER INTO TEXT
 7469 011174' 001000  000002    42740           STAX    B               ;STORE A ZERO IF THE END
 7470 011175' 001000  000510    42760           RZ                      ;ZERO, END OF LINE,
 7471 011176' 001000  000362    42780           JP      PLOOP           ;REGULAR CHAR, JUST PRINT IT
 7472 011177' 000000  011170'
 7473 011200' 000000  011164'
 7474 011201' 001000  000376    42800           CPI     ELSETK          ;IF ITS "ELSE" DON'T PRINT THE COLON
 7475 011202' 000000  000020
 7476                           42820                                   ;IN FRONT OF IT
 7477 011203' 001000  000310    42840           CZ      DCXBRT##        ;BACKUP STUFF COUNT TO ELIMINATE
 7478 011204' 000000  000000#
 7479 011205' 000000  011177'
 7480 011206' 001000  000326    42860           SUI     127             ;GET RID OF SIGN BIT AND ADD ONE
 7481 011207' 000000  000177
 7482 011210' 001000  000345    42880           PUSH    H               ;SAVE CURRENT POSIT
 7483 011211' 001000  000021    42900           LXI     D,RESLST        ;GET RESLST POINTER,
 7484 011212' 000000  000172'
 7485 011215' 000000  011204'
 7486 011214' 001000  000325    42920   RESRCH: PUSH    D
 7487 011215' 001000  000365    42940           PUSH    PSW             ;SAVE THE RESERVED WORD NUMBER
 7488 011216' 001000  000032    42960   RESCR1: LDAX    D               ;GET CHARACTER FROM RESLST
 7489 011217' 001000  000023    42980           INX     D               ;BUMP RESLST POINTER
 7490 011220' 001000  000267    43000           ORA     A               ;TEST BITS
 7491 011221' 001000  000362    43020           JP      RESCR1          ;NOT AT END OF RESERVED WORD YET
 7492 011224' 000000  011216'
 7493 011223' 000000  011214'
 7494 011224' 001000  000361    43040           POP     PSW
 7495 011225' 001000  000075    43060           DCR     A               ;DECREMENT CHAR
 7496 011226' 001000  000341    43080           POP     H               ;POP START POINTER HERE
 7497 011227' 001000  000302    43100           JNZ     RESRCH          ;NOT AT END OF RESLST YET,
 7498 011230' 000000  011214'
 7499 011231' 000000  011222'
 7500                           43120   ;HERE WHEN FOUND RIGHT RESERVED WORD
 7501 011236' 001000  000176    43140   PRIT3:  MOV     A,M             ;GET A CHARACTER FROM RESERVED WORD
```

```
7502  011235' 001000  000267          43160           ORA     A               ;SET CONDITION CODES
7503  011236' 001000  000002          43180           STAX    B
7504  011237' 001000  000372          43200           JM      PRIT4
7505  011236' 000000  011167'
7506  011231' 000000  011230'
7507  011240' 001000  000003          43220           INX     B
7508  011241' 001000  000043          43240           INX     H               ;BUMP RESLST POINTER
7509  011242' 001000  000303          43260           JMP     PRIT3           ;PRINT THE REST
7510  011245' 000000  011236'
7511  011244' 000000  011236'
7512                                   43280   ;
7513                                   43300   ; THE FOLLOWING CODE IS FOR THE DELETE RANGE
7514                                   43320   ; COMMAND, BEFORE THE LINES ARE DELETED, 'OK'
7515                                   43340   ; IS TYPED.
7516                                   43360   ;
7517  011245' 001000  000315          43380   DELETE: CALL    SCNLIN          ;SCAN LINE RANGE
7518  011246' 000000  002333'
7519  011247' 000000  011245'
7520  011250' 001000  000321          43400           POP     D               ;POP MAX LINE OFF STACK
7521  011251' 001000  000305          43420           PUSH    B               ;SAVE POINTER TO START OF 1ST LINE
7522  011252' 001000  000315          43440           CALL    FNDLIN          ;FIND THE LAST LINE
7523  011255' 000000  002371'
7524  011254' 000000  011246'
7525  011255' 001000  000301          43460           POP     B               ;GET POINTER TO FIRST IN [B,C]
7526  011256' 001000  000345          43480           PUSH    H               ;SAVE THE POINTER TO THE NEXT LINE
7527  011257' 001000  000041          43500           LXI     H,READY         ;PRINT "OK" PREMATURELY
7528  011260' 000000  001725'
7529  011261' 000000  011252'
7530  011262' 001000  000315          43520           CALL    STROUT
7531  011263' 000000  007743'
7532  011264' 000000  011260'
7533  011265' 001000  000041          43540           LXI     H,FINI          ;GO BACK TO FINI WHEN DONE
7534  011266' 000000  002276'
7535  011267' 000000  011263'
7536  011270' 001000  000343          43560           XTHL                    ;[H,L]=POINTER TO THE NEXT LINE
7537  011271' 001000  000353          43580   DEL:    XCHG                    ;[D,E] NOW HAVE THE POINTER TO THE LINE
7538                                   43600           ;BEYOND THIS ONE
7539  011272' 001000  000052          43620           LHLD    VARTAB          ;COMPACTIFYING TO VARTAB
7540  011273' 000000  001621'
7541  011274' 000000  011266'
7542  011275' 001000  000032          43640   MLOOP:  LDAX    D
7543  011276' 001000  000002          43660           STAX    B               ;SHOVING DOWN TO ELIMINATE A LINE
7544  011277' 001000  000003          43680           INX     B
7545  011300' 001000  000023          43700           INX     D
7546  011301' 001000  000347          43720           COMPAR
7547  011302' 001000  000322          43740           JNC     MLOOP           ;DONE COMPACTIFYING?
7548  011305' 000000  011275'
7549  011304' 000000  011273'
7550  011305' 001000  000140          43760           MOV     H,B
7551  011306' 001000  000151          43780           MOV     L,C
7552  011307' 001000  000043          43800           INX     H               ;NEW VARTAB
7553  011310' 001000  000042          43820           SHLD    VARTAB
7554  011311' 000000  001621'
```



```
7555  011312' 000000  011303'
7556  011315' 001000  000311          43840           RET>
7557                                   43860   PAGE
```

```
 7558                                   43880   SUBTTL  DISK CODE
 7559                                   43900   IFN     DSKFUN,<
 7560                                   43920   ;
 7561                                   43940   ; THE STATEMENT DSKO$ STRING,SECTOR WRITES
 7562                                   43960   ; THE STRING (UP TO 132 DECIMAL CHARS
 7563                                   43980   ; ON THE SECTOR SPECIFIED,
 7564                                   44000   ; DSK$ (SECTOR) IS A STRING FUNCTION THAT
 7565                                   44020   ; RETURNS THE 133 BYTE STRING STORED ON SECTOR,
 7566                                   44040   ;
 7567                                   44060   DSKO$:  CALL    FRMEVL          ;EVALUATE FORMULA
 7568                                           SYNCHK  44              ;FOLLOWED BY COMMA
 7569                                   44100           PUSH    H               ;SAVE TEXT POINTER
 7570                                   44120           CALL    FRESTR          ;FREE UP THE PACKD
 7571                                   44140           XTHL                    ;[M,L]=TXTPTR SAVE POINTER AT
 7572                                   44160                                   ;STRING DESCRIPTOR ON THE STACK
 7573                                   44180           CALL    GETBYT          ;EVALUATE 2ND ARG(SECTOR) IN (E)
 7574                                   44200           XTHL                    ;SAVE TEXT POINTER, GET DESC,
 7575                                   44220           PUSHM                   ;[C]=LENGTH [M,L]=POINTER
 7576                                   44240           PUSHM
 7577                                   44260           POP     H               ;[M,L] GET STRING POINTER
 7578                                   44280           POP     B
 7579                                   44300           MOV     B,A             ;SECTOR NUMBER INTO [B]
 7580                                   44320           MVI     A,^D137
 7581                                   44340           SUB     C
 7582                                   44360           JC      FCERR           ;STRING TOO LONG
 7583                                   44380           INR     A
 7584                                   44400           MOV     E,A             ;NUMBER OF ZEROS+1
 7585                                   44420           MVI     D,84            ;SET UP A MASK
 7586                                   44440           INR     C
 7587                                   44460           MVI     A,4             ;LOAD THE HEAD
 7588                                   44480           OUT     9               ;TO DISK STATUS
 7589                                   44500   SECLP:  IN      ^O11            ;GET SECTOR STATUS
 7590                                   44520           RAR                     ;TEST FOR START OF SECTOR
 7591                                   44540           JNC     SECLP           ;KEEP WAITING
 7592                                   44560           ANI     63              ;START OF SECTOR, RIGHT ONE
 7593                                   44580           CMP     B               ;COMPARE TO FIND OUT
 7594                                   44600           JNZ     SECLP           ;IF NOT
 7595                                   44620           MVI     A,128           ;WRITE ENABLE DISK
 7596                                   44640           OUT     9
 7597                                   44660           MVI     B,255           ;ALL ONE'S ALWAYS WRITTEN FIRST
 7598                                   44680   WRITOK: IN      8               ;GET STATUS
 7599                                   44700           ANA     D               ;WRITE OK
 7600                                   44720           JZ      WRITOK          ;NO, MORE LOOPING,
 7601                                   44740           MOV     A,B             ;GET CHARACTER TO WRITE
 7602                                   44760           OUT     10              ;SEND IT OUT
 7603                                   44780           DCR     C               ;TEST FOR NULL
 7604                                   44800           JZ      ZKLOP
 7605                                   44820   NOTYTO: IN      8               ;POLL
 7606                                   44840           ANA     D               ;MASK TEST
 7607                                   44860           JZ      NOTYTO          ;WAITING
 7608                                   44880           MOV     A,M             ;GET CHARACTER
 7609                                   44900           OUT     10
 7610                                   44920           DCR     C               ;DECREMENT CHARACTER COUNT
```



```
 7611                                   44940           INX     H
 7612                                   44960           JNZ     NOTYTO
 7613                                   44980   ZKLOP:  IN      8
 7614                                   45000           ANA     D
 7615                                   45020           JZ      ZKLOP
 7616                                   45040           XRA     A               ;PUT OUT A ZERO
 7617                                   45060           OUT     10
 7618                                   45080           DCR     E
 7619                                   45100           JNZ     ZKLOP
 7620                                   45120   TRUFIN: MVI     A,8             ;UNLOAD THE HEAD
 7621                                   45140           OUT     9
 7622                                   45160           POP     H
 7623                                   45180           RET                     ;DONE
 7624                                   45200
 7625                                   45220   DSKI$:  MVI     A,137           ;A LOT OF CHARACTERS ARE COMING
 7626                                   45240           CALL    $TRINI          ;MAKE ROOM!
 7627                                   45260           CALL    CONINT          ;WHERE ARE THEY?
 7628                                   45280                                   ;SECTOR NOW IN (E)
 7629                                   45300           LXLD    DSCTMP+2        ;PLACE TO STORE THEM
 7630                                   45320           MVI     A,4             ;LOAD THE HEAD
 7631                                   45340           OUT     9
 7632                                   45360   SECLP2: IN      9               ;GET SECTOR INFO
 7633                                   45380           ORA     A               ;SEE IF BEGINNING OF SECTOR(READ)
 7634                                   45400           JP      SECLP2          ;IF NOT, KEEP WAITING
 7635                                   45420           RAR                     ;FIX UP SECTOR #
 7636                                   45440           ANI     63              ;GET SECTOR #
 7637                                   45460           CMP     E               ;IS IT THE ONE WE WANTED
 7638                                   45480           JNZ     SECLP2          ;TRY TO FIND IT
 7639                                   45500           MVI     C,137           ;GET # OF CHARS TO READ
 7640                                   45520   READOK: IN      8               ;GET DISK STATUS
 7641                                   45540           ORA     A               ;READY TO READ BYTE
 7642                                   45560           JP      READOK
 7643                                   45580           IN      10              ;READ THE STUFF
 7644                                   45600           MOV     M,A             ;SAVE IN STR
 7645                                   45620           INX     H               ;BUMP DEST POINTER
 7646                                   45640           DCR     C               ;LESS CHARS
 7647                                   45660           JNZ     READOK
 7648                                   45680           MVI     A,8             ;UNLOAD HEAD
 7649                                   45700           OUT     9
 7650                                   45720           JMP     FINBCK          ;USE CHRS TO FINISH UP
 7651                                   45740
 7652                                   45760   PATCH:  BLOCK   28>
 7653                                   45780           PAGE
```

```
                                45800   SUBTTL  CLOAD,CSAVE,CONSOLE
                                45820   ;
                                45840   ; THE CONSOLE COMMAND ALLOWS THE USER TO CHANGE THE I/O CHANNEL
                                45860   ; THAT THE USER TERMINAL IS ON, BY GIVING THE COMMAND CONSOLE X
                                45880   ; WHERE X IS SOME INTEGER THE TERMINAL DEVICE WILL BE POLLED FROM
                                45900   ; CHANNELS X AND X+1. RESTARTING AT LOCATION ZERO FORCES THE TERMINAL
                                45920   ; TO BE ON CHANNEL ZERO AGAIN.
                                45940   ;
                                45960   IFN     CONSSW,<
                                45980   INTERNAL        CONSOC
                                46000   CONSOO: XRA     A               ;FORCE A CHANNEL ZERO CONSOLE
                                46020           CALL    CONS2           ;ON RESTART AT ZERO
                                46040           JMP     READY          ;TYPE "OK" AND ACCEPT INPUT
                                46060   CONSOL: CALL    GETBYT          ;FETCH AN INTEGER INTO [A]
                                46080           RNZ                     ;CHECK FOR A TERMINATOR
                                46100   CONS2:
                                46120   IFN     REALIO,<
                                46140           STA     CHLCA1          ;CHANGE ALL THE FLAG INPUT CHANNEL REFERENCES
                                46160           STA     CHLCAE
                                46180           STA     CHLCA3>
                                46200   IFN     LENGTH,<
                                46220           STA     CHLCA4>
                                46240           INR     A               ;[A]=DATA INPUT CHANNEL
                                46260           STA     CHLCB1          ;CHANGE ALL THE DATA INPUT CHANNEL REFERENCES
                                46280           STA     CHLCB2
                                46300           RET>
                                46320   IFN     CASSW,<
                                46340   ;
                                46360   ; CASIN READS A CHARACTER FROM THE CASSETTE
                                46380   ; INTO [A] WITHOUT MODIFYING ANYTHING BUT [A] AND THE CONDITION
                                46400   ; CODES
                                46420   ;
                                46440   CASIN:  IN      6               ;ROUTINE TO READ A CHARACTER
                                46460           ANI     IDONE           ;FROM THE CASSETTE INTO [A]
                                46480           JNZ     CASIN
                                46500           IN      7               ;READ THE DATA
                                46520           RET
                                46540   ;
                                46560   ; CASOUT OUTPUTS THE CHARACTER IN [A] TO THE CASSETTE
                                46580   ; WITHOUT MODIFYING ANYTHING
                                46600   ;
                                46620   TWOCSO: CALL    CASOUT          ;DOUBLE OUT OF [A]
                                46640   CASOUT: PUSH    PSW             ;ROUTINE TO WRITE A CHARACTER IN [A]
                                46660   CASLK:  IN      6               ;ONTO THE CASSETTE
                                46680           ANI     ODONE
                                46700           JNZ     CASLK           ;WAIT TILL CASSETTE IS READY
                                46720           POP     PSW             ;GET THE CHARACTER BACK
                                46740           OUT     7               ;OUTPUT THE CHARACTER
                                46760           RET
                                46780   ;
                                46800   ; THE CSAVE COMMAND WRITES A PROGRAM ONTO CASSETTE BY DUMPING
                                46820   ; BASICS CORE, THE HEADER IS THREE 211'S FOLLOWED BY A ONE
                                46840   ; CHARACTER FILE NAME, THE END IS THREE ZEROS IN A ROW.
```



```
                                46860   ;
                                46880   CSAVE:  PUSH    H
                                46900           MVI     A,211
                                46920           CALL    CASOUT          ;PUT OUT THE START BYTES
                                46940           CALL    TWOCSO          ;TWO MORE TIMES
                                46960           MOV     A,M             ;GET FILENAME
                                46980           CALL    CASOUT          ;STORE AFTER 211'S
                                47000           LHLD    TXTTAB          ;START OF PROGRAM
                                47020           XCHG
                                47040           LHLD    VARTAB          ;END OF PROGRAM
                                47060   LOPCSO: LDAX    D               ;GET A BYTE FROM THE PROGRAM
                                47080           INX     D
                                47100           CALL    CASOUT          ;SEND IT OUT TO THE CASSETTE
                                47120           COMPAH                  ;THE END?
                                47140           JNZ     LOPCSO          ;IF NOT,OUTPUT MORE
                                47160           CALL    TWOCSO          ;TWO MORE 0'S TO MARK THE END
                                47180           POP     H               ;RESTORE THE TEXT POINTER
                                47200           CHRGET                  ;GO PAST THE FILE NAME
                                47220           RET
                                47240   ;
                                47260   ; THE CLOAD COMMAND CLEARS CORE AND THEN READS A PROGRAM
                                47280   ; FROM CASSETTE, SINCE THE LINKS OF THE FILE ON CASSETTE
                                47300   ; WILL BE WRONG IF THE FILE WAS SAVED WITH A DIFFERENT VERSION OF
                                47320   ; BASIC FINI IS JUMPED TO, A SCRATCH IS DONE AT THE START SO RESTARTS
                                47340   ; AT 0 WON'T LEAVE THINGS IN A GARBAGE STATE.
                                47360   ;
                                47380   CLOAD:  STA     FACLO           ;SAVE THE FILENAME
                                47400           CALL    SCRTCH          ;RESET EVERYTHING
                                47420   LOPCLK: MVI     B,3             ;NUMBER OF START CHARACTERS
                                47440   LOPCL2: CALL    CASIN           ;GET A CHARACTER
                                47460           CPI     211             ;START CHARACTER?
                                47480           JNZ     LOPCLK          ;NO, RESET COUNT AND LOOK SOME MORE
                                47500           DCR     B               ;DECREMENT THE COUNT
                                47520           JNZ     LOPCL2          ;BEEN THREE YET?
                                47540           LXI     H,FACLO         ;POINT AT THE FILENAME
                                47560           CALL    CASIN           ;READ THIS FILENAME
                                47580           CMP     M               ;THE RIGHT FILE?
                                47600           JNZ     LOPCLK          ;IF NOT,START COMPLETELY OVER
                                47620           LHLD    TXTTAB          ;PLACE TO STORE THE PROGRAM
                                47640   DOCKS:  MVI     B,4             ;NUMBER OF ZEROS TO GET
                                47660                                   ;BEFORE STOPPING
                                47680   DOCSMR: CALL    CASIN           ;GET A CHARACTER
                                47700           MOV     M,A             ;STORE IT
                                47720           CALL    REASON          ;MAKE SURE THERE IS ROOM
                                47740           MOV     A,M             ;REGET THE CHARACTER
                                47760           ORA     A               ;A ZERO?
                                47780           INX     H
                                47800           JNZ     DOCKS           ;RESET # OF ZEROS SEEN
                                47820           DCR     B               ;DECREMENT NUMBER OF ZEROS
                                47840           JNZ     DOCSMR          ;SEEN FOUR?
                                47860           SHLD    VARTAB          ;SET UP END OF PROGRAM
                                47880           LXI     H,READY         ;TYPE "OK" PREMATURELY
                                47900           CALL    STRCUT
```

```
BASIC MC8 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(113) 03:12 10-SEP-75 PAGE 29-2
F3      NAC     6-SEP-64 03:11          CLOAD,CSAVE,CONSOLE

  7760                                  47920           JMP     FINI>           ;FIX UP THE LINKS AND GO BACK TO MAIN
  7761                                  47940   PAGE
```

```
BASIC MC8 8080  GATES/ALLEN/DAVIDOFF    MACRO 47(113) 03:12 10-SEP-75 PAGE 30
F3      NAC     6-SEP-64 03:11          PEEK AND POKE

  7762                                  47960   SUBTTL PEEK AND POKE
  7763                                  47980   IFN     LENGTH-2,<
  7764                                  48000   IFE     LENGTH-2,<
  7765  011314' 001000  000315         48020   PEEK:   CALL    FNCINT          ;GET AN INTEGER IN (H,L)
  7766  011315' 000000  011023'
  7767  011316' 000000  011311'
  7768  011317' 001000  000176         48040           MOV     A,M>            ;GET THE VALUE TO RETURN
  7769                                  48060   IFN     LENGTH-2,<
  7770                                  48080   PEEK:   CALL    POSINT          ;GET THE VALUE OF FACLO INTO (D,E)
  7771                                  48100           LDAX    D>              ;READ THE VALUE
  7772  011320' 001000  000304         48120           JMP     SNGFLT          ;AND FLOAT IT
  7773  011321' 000000  007400'
  7774  011322' 000000  011315'
  7775                                  48140   IFE     LENGTH-2,<
  7776  011323' 001000  000315         48160   POKE:   CALL    FNMEVL
  7777  011324' 000000  006330'
  7778  011325' 000000  011321'
  7779  011326' 001000  000345         48180           PUSH    H               ;SAVE THE TEXT POINTER
  7780  011327' 001000  000315         48200           CALL    FNCINT          ;GET INTEGER VALUE OF FAC IN (H,L)
  7781  011330' 000000  011315'
  7782  011331' 000000  011324'
  7783  011332' 001000  000343         48220           XTHL>                   ;GET BACK THE TEXT POINTER
  7784                                  48240   IFN     LENGTH-2,<
  7785                                  48260   POKE:   CALL    INT102          ;READ LOCATION TO POKE
  7786                                  48280           PUSH    D>              ;SAVE THE LOCATION
  7787  011333' 001000  000317         48300           SYNCHK  44              ;CHECK FOR A COMMA
  7788  011334' 000000  000054
  7789  011335' 001000  000315         48320           CALL    GETBYT
  7790  011336' 000000  011020'
  7791  011337' 000000  011330'
  7792  011340' 001000  000321         48340           POP     D               ;GET THE ADDRESS BACK
  7793  011341' 001000  000022         48360           STAX    D               ;STORE IT AWAY
  7794  011342' 001000  000311         48380           RET>                    ;SCANNED EVERYTHING
  7795                                  48400   ;
  7796                                  48420   ; NOTE: IN THE 8K PEEK ONLY ACCEPTS POSITIVE NUMBERS UP TO 32767
  7797                                  48440   ; POKE WILL ONLY TAKE AN ADDRESS UP TO 32767 , NO
  7798                                  48460   ; FUDGING ALLOWED, THE VALUE IS UNSIGNED.
  7799                                  48480   ;
  7800                  011336'         48500   .CI=.P
  7801                                  48520   END

NO ERRORS DETECTED

PROGRAM BREAK IS 011343

10K CORE USED
```

A	000007		CMLCA1	003114'	SIN	DCOMP	000702'	EXT
ABS	000147'	EXT	CMLCA2	003127'	SIN	DCXSRT	011204'	EXT
AFF	000104		CMLCA3	003341'	SIN	DDIV	000700'	EXT
ALLLST	002354'		CMLCA4	003303'	SIN	DEF	007411'	
ALLSTK	010570'		CMLCB1	003124'	SPD	DEFFIN	007521'	
ANDCKD	005571'		CMLCB2	003136'	SPD	DEL	011271'	
APPLOP	005642'		CMTCON	003465'		DELETE	011245'	
ARG	001654'	INT	CMTWFL	001541'	INT	DFALLO	001633'	INT
ARGLV	001645'	INT	CULIS	002705'		DIM	006500'	
ARYSTK	010177'		CUMPRT	004553'		DIMCON	006473'	
ARYTAB	001623'		CUNIA	006405'	EXT	DIMFLG	001542'	
ARYVA2	010137'		CONINT	011023'		DIRIB	003524'	
ARYVAR	010140'		CUNSIM	006041'	EXT	DMULT	000676'	EXT
ASC	010515'		CONSSW	000000	SPD	DNTCPY	004217'	
ASPA2	004626'		CONT	003542'		DOASIG	005056'	
ASPAC	004627'		CONTRW	000001	SPD	DOCMP	006375'	
ATN	000137'	EXT	CONTW	000017		DOCONO	004554'	
ATNFIX	000137'	INT	COPNUM	004237'		DODSP	005756'	
ATNTK	000300	SPD	COS	000131'	EXT	DOMIN	006146'	
B	000000		COSFIX	000131'	INT	DOMMUL	007326'	
BLTLOP	002013'		CR	000015	SPD	DORES	001544'	
BLTU	002005'		CRDD	004523'	INT	DOSIZT	004621'	
BLTUC	002010'		CRDONE	002753'		DSCTMP	001570'	
BLTUCN	000016	SPD	CRFIN	004534'		DSKFUN	000000	SPD
BRKTXT	001734'		CRUNCH	002533'		DSUB	000674'	EXT
BSERR	007677'	INT	CSLOOP	006345'		DVBERN	002675'	INT
BUF	001431'	INT	CURLIN	001607'	INT	DVAR	010214'	
BUFLEN	000110	SPD	CZLOOP	002316'		DVAR2	010213'	
BUFLIN	011163'		D	000002		DVARB	010214'	
BUFMIN	001430'		DADD	000672'	EXT	E	000003	
C	000001		DANDJM	006437'		EATEM	006535'	
CASSW	000000	SPD	DATA	004072'		EDIT	000632'	EXT
CAT	010320'		DATATK	000203	SPD	ELSE	004070'	
CHEAD	002302'		DATBK	005025'		ELSETK	000220	SPD
CHKLOP	004560'		DATFND	005166'		END	003470'	
CHKSTR	010432'	EXT	DATLIN	001577'		ENDCON	003501'	
CHRB	010532'		DATLOP	005163'		ENDREL	005415'	
CHRCON	003433'		DATPTR	001627'		ENDTK	000200	SPD
CHRGTR	003426'		DATSNE	002064'		EQULTK	000260	
CLEAR	003703'		DBLOOP	000672'		ERRBS	000011	SPD
CLEARC	002443'					ERRCN	000021	SPD
CLMBLC	000016	SPD				ERRDD	000012	SPD
						ERRDIM	007532'	
						ERRDV0	000013	SPD
						ERRFC	000005	SPD
						ERRFIN	002127'	
						ERRID	000014	SPD
						ERRLS	000017	SPD
						ERRNF	000001	SPD
						ERROD	000004	SPD

ERROM	000007	SPD	FPWR	000000	EXT	INCHR	003126'	
ERRUR	002102'	INT	FPWRQ	005562'	EXT	INDLOP	006754'	
ERRUV	000006	SIN	FRCOBL	005751'	EXT	INEG	000000	EXT
ERRXG	000003	SPD	FRCINT	011330'	EXT	INIT	002163'	EXT
ERRSN	000002	SPD	FRCSNG	007506'	EXT	INLIN	002776'	
ERRSO	000016	SPD	FRCSTR	000000	EXT	INLINC	003003'	
ERRST	000022	SPD	FRCTBL	000664'		INLINN	002772'	
ERRTAB	000730'		FRE	007337'		INLPNM	007252'	
ERRTK	000015	SIN	FREFAL	010434'		INPCON	004157'	
ERRUF	000022	SPD	FRLSTK	010431'		INPCON	004765'	
ERRUS	000010	SPD	FRETM2	010437'		INPRT	002141'	EXT
EVAL	006061'		FRETMP	010440'		INPUT	004711'	
EXCHQT	004100'		FRETOP	001575'	INT	INPWRD	010720'	
EXIGNT	005142'		FRMCHK	005337'		INRART	000000	EXT
EXP	000127'	EXT	FRMEVL	005336'		INT	000000	EXT
EXPSTK	005533'		FSUB	000706'	EXT	INTOSP	000716'	
EXTFNC	000001	SPD	FUNCTB	000001	SPD	INTIO2	003623'	
FAC	001642'	INT	FUNOSP	000103'		INTIOX	003622'	
FACDBL	005775'		GARBAG	010042'		INTXT	001720'	INT
FACLO	001637'	INT	GARBAG	010026'		INXHRT	002664'	EXT
FACSNG	000034'		GETAGN	004741'		ISARY	006745'	
FADD	000704'	EXT	GETBCU	007752'	EXT	ISCNTC	003460'	
FADJS	005260'	EXT	GETBYT	011020'		ISFUN	006204'	
FALSIF	004363'		GETDEF	007245'		ISGOSB	004306'	
FBUFFR	001655'	INT	GETFNM	007550'		ISLIN	000020'	EXT
FCERR	010776'	INT	GETSPA	007772'		ISLET	003612'	
FCOMP	005274'	EXT	GETSTK	002024'		ISSEC	006534'	
FDIV	000712'	EXT	GETYPE	006307'	INT	ISUB	000720'	EXT
FIN	011064'	EXT	GIVDBL	007372'		ISVAR	006164'	
FINBCK	010546'		GIVINT	007402'		KLOOP	002544'	
FINGO	006276'		GONE	003370'		L	000005	
FINI	002276'		GONE2	003376'		LABBCK	006162'	
FINILL	006516'		GONE3	003375'		LABNUM	000306	SPD
FINNOW	007331'		GOODCH	003044'		LEFT$	010552'	
FINPTR	006742'		GOSUB	003770'		LEFT2	010560'	
FINREL	005605'		GOSUTK	000214		LEFT3	010556'	
FINIMP	005544'		GOTO	004010'		LEN	010581'	
FLGINP	001602'		GOTOTK	000210	SPD	LEN1	010505'	
FLOAT	000000	EXT	GRBPAS	010254'		LENGTH	000002	SPD
FLOATR	000000	EXT	GREATK	000257		LEPSKP	002117'	
FMULT	000710'	EXT	GTBYTC	011017'				
FNDFOR	001704'		H	000004				
FNDLIN	002371'		HAVTYP	006574'				
FNDUER	007443'		IADJ	000716'	EXT			
FNDVAR	010045'		ICOMP	000726'	EXT			
FNINP	010712'		IDIV	000724'	EXT			
FNOUT	010725'		IDONE	000001	SPD			
FNTK	000243	SPD	IF	004325'				
FNWAIT	010733'		IFOKDN	003267'				
FOR	003154'		IFTK	000212	SPD			
FORTK	000201	SPD	ILLFUN	010776'	INT			
FOUND	002646'		IMULT	000722'	EXT			
FOUT	007565'	EXT						

```
LESSTK          000201          MOVMM           010154' EXT     OMERR           002057' INT
LET             004131'         MOVSTK          010417'         ONEFUN          000262
LINCHK          004475'         MULLIM          000001  SPD     ONELIN          002365'
LINGET          003642' INT     MUSTCM          002617'         ONEUN           003265'
LINGT2          003643'         NEG             000000  EXT     ONGOTO          004271'
LINLEN          000110  SPD     NEWCMK          004427'         OPRTYP          001544'
LINLIN          002764'         NEWSTT          003342'         OPTAB           000163'
LINPRT          011134' EXT     NEXT            005225'         OKFIN           006463'
LINPT1          003183' SIN     NEXTC           005230'         OUTDEL          003061'
LINPT2          003574' SIN     NFERR           002100'         OUTCON          003065'
LINPT3          004474' SIN     NMAHY1          007051'         OUTGO           000030  INT
LINPT4          004537' SIN     NMAHY2          007058'         OUTWRD          010736'
LIST            011072'         NNREL           000003  SPD     PARCMK          006136'
LIST4           011077'         NODATT          002716'         PEEK            011314'
LISTEN          000001  SPD     NODEL           002227'         PLOOP           011170'
LOG             000125' EXT     NOPRIN          003113'         PLUSTK          000250  INT
LOOP            002374'         NOSEC           006547'         POKE            011323'
LOOPEN          005320'         NOTABK          004637'         POPGOF          001774'
LOOPER          001750'         NOT.IM          007152'         POPNRT          000000  EXT
LOOPON          004307'         NOTER           006412'         POS             007406'
LOPUT2          004773'         NOTFOU          007104'         PPGWRT          010024'
LOPUTA          007015'         NOTFNS          006671'         PREAM           010764'
LOPUTM          006623'         NOTFNT          006272'         PREAM2          010767'
LOPUTT          007135'         NOTIT1          006660'         PRINT           004411'
LOPUTL          005363'         NOTLST          010477'         PRINTC          004414'
LPOPEN          005341'         NOTUL           003176'         PRINTK          000231  SPD
LPTLEN          000110  SPD     NOTUT1          004735'         PRIT3           011232'
LPTSW           000000  SPD     NOTSTR          005360'         PRLT4           011167'
LSTOPK          000007  SPD     NOTTMR          010753'         PRTNUL          004537'
M               000006  SPD     NOTTIK          000246  SPD     PSW             000006  SPD
MAIN            002165'         NOTTRC          003567'         PTRGET          006505'
MAKINT          007404' EXT     NOWGET          005053'         PTRGT2          006512'
MEMSIZ          001545' INT     NUMLIN          005214'         PUFOUT          000000  EXT
MID             010043'         NTMIS           002736'         PUNE            000000  SPD
MID2            010065'         NTMIS1          002742'         PUSMF           006036' EXT
MINUTK          000E51  INT     NULENT          000046'         PUSMMA          000100'
MLOOP           011275'         NULL            003566'         PUTDE1          007324'
MLOOPR          002266'         NUMCMD          000040          PUTNEW          007705'
MORCOM          004566'         NUMFMT          000051  SPD     PUTVAL          004261'
MORLIN          003646'         NUMINS          005070'         Q               000022
MORPW           004410'         NUMLEV          000025  SPD     QINLIN          002522' INT
MOVE            000000  EXT     NUMREL          005476'         QINT            000000  EXT
MOVFM           005253' EXT     NUMTMP          000005  SPD
MOVTM           005746' EXT     NUTPOS          000070  SPD
MOVINS          010407'         NXTCON          003276'
MOVLP           010420'         NXTRES          000625'
MOVMF           007476' EXT     OUCNE           000200  SPD
MOVMF           006004' EXT     OKGOTO          004386'
                                OKNORM          006251'
                                OLDLIN          001611'
                                OLDTXT          001613'
```



```
RAMBOT          020000          SNGLO           006026'         TABER           004577'
READ            004760'         SNGOSP          007004'         TABTK           000240
READY           002145' INT     SNGFLT          007400'         TAN             000135' EXT
REALIO          000001  SPD     SP              000006          TANFIX          000135' INT
REASON          002E45' INT     SPCTK           000242  SPD     TEMP            001603'
REDOY           001725'         SQR             000121' EXT     TEMP2           001605' INT
REDINP          004136'         SQRFIX          000121'         TEMP3           001575' INT
REM             004074'         SQRTK           000271  SPD     TEMP6           007124'
REMEM           004103'         STAINP          010754'         TEMPPT          001547' INT
REMTK           000216          STAKT           000000'         TEMPST          001551' INT
REPINI          002102' INT     STEPTK          000247          THENTK          000245
REPOUT          004632'         STKOOL          005725'         TMEMR           005703' EXT
RESCK1          011216'         STKIN1          002470'         TOFF            003605'
RESER           002027'         STKSNG          006021'         TON             003604'
RESFIN          003453'         STKTOP          001615' INT     TOPLON          000035  SPD
RESLST          000172'         STMDSP          000564'         TOTK            000241  SPD
RESMEM          011214'         STOP            003472'         TRCFLG          001631'
RESTOR          003446'         STPEND          003580'         TRMNON          004667'
RETADP          005358'         STPRDY          002144'         TRMOK           005111'
RETURN          004044'         STRI            007564'         TRYAGN          004640'
RIGHT$          010631'         STRI            002716'         TRYGIV          007774'
RND             000123' EXT     STRAD1          007632'         TRYIN           003126'
RNDFIX          000123'         STRAD2          007627'         TRYOUT          003113'
RUN             003754'         STRCMP          006320'         TSTACK          000000  EXT
RUNC            002457'         STRCPY          007601'         TSTOP           005357'
RUNL2           004097'         STRUN2          005077'         TTICMN          000021  SPD
SCNLIN          002333'         STRLON          004507'         TTOCMN          000021  SPD
SCRATH          002421'         STRLNU          001625' INT     TTYCMN          003070'
SCRATK          000257          STRFIN          007665'         TTYIST          004616'
SCRTCH          002462' INT     STRGET          007646'         TTYPOS          000047'
SETUBL          005750'         STRING          000021  SPD     TVAR            010063'
SETUP           011003'         STRIN1          007624'         TXTTAB          001617' INT
SGN             000103' EXT     STRLIT          007637' INT     UMULT           007267' EXT
SIGN            000050' INT     STRLT2          007643'         USERR           004037'
SIGNC           000055' EXT     STRLT3          007642'         USINTK          000240  SPD
SIGNS           005373' EXT     STRLT1          007640'         USRLOC          000111'
SIN             000133' EXT     STRNG           002721'         VAL             011002'
SINFIX          000133' INT     STROUI          007742' INT     VALINT          000000  EXT
SKPMMF          004365'         STROUT          007743' INT     VALSNG          000000  EXT
SKPVAR          010130'         STRPR2          007755'         VALTYP          001543' INT
SMLVAL          007322'         STRPRT          007746'         VARTAB          001621'
SNEMM           006072' INT     SIRSIZ          000023  SPD     VINT            000105' EXT
SNGOML          005745'         SUFFM           002667'         VMOVAF          005777' EXT
                                SUBFLG          001601'         VMOVE           007724' EXT
                                SVAR            010122'         VMOVFA          000020  EXT
                                SVARS           010077'         VMOVFM          006200' EXT
                                TABEND          002653'         VMOVMF          004264' EXT
                                                                VNEG            006160' EXT
                                                                VPLSMO          005536'
                                                                VSIGN           004347' EXT
```

```
ZERITA          007207'
ZERO            011046' EXT
ZEROEK          006723'
SCOLE           000000' INT
```

A	764	791	812	827	830	852	,740	1756	1796	1807	1922	1924	1943	1965
	1966	1978	2049	2052	2069	2078	2154	2187	2189	2269	2272	2298	2299	2305
	2312	2320	2321	2322	2330	2335	2368	2373	2376	2417	2418	2419	2519	2520
	2523	2528	2606	2732	2803	2806	2875	2882	2887	2907	2912	2918	2944	2962
	2970	2979	3020	3026	3041	3070	3089	3103	3112	3152	3153	3197	3220	3222
	3225	3225	3397	3399	3402	3401	3428	3527	3534	3554	3556	3561	3587	3588
	3599	3623	3721	3834	3834	3859	3866	3872	3880	3953	3972	3973	3974	4019
	4041	4080	4081	4096	4126	4141	4149	4161	4182	4249	4303	4316	4392	4453
	4535	4555	4566	4567	4571	4583	4618	4662	4713	4773	4823	4827	4841	4842
	4848	49.1	4916	4919	5123	5127	5193	5241	9242	5243	5349	5370	5373	5374
	5377	5379	5396	5397	540E	5440	5442	5443	5445	5473	5478	5479	5481	5487
	5486	5490	5529	5540	5541	5559	5594	5622	5628	5643	5664	5675	5895	5896
	5897	5942	5948	5965	6017	6019	6052	6053	6100	6104	6108	6181	6283	6285
	6286	6289	6270	6271	6423	6437	6444	6402	6497	6511	6540	6542	6562	6588
	6612	6659	6671	6694	6746	6759	6760	6782	6825	6871	6929	6976	6981	7052
	7070	7071	7074	7075	7092	7101	7126	7128	7130	7135	7161	7191	7218	7223
	7224	7227	7342	7343	7351	7368	7380	7409	7439	7465	7467	7490	7495	7501
	7502	7768												
ABS	904	904#												
AFFA	986#	986	987#	987	988#	988	989#	989	990#	990	991#	991	992#	992
	993#	993	994#	994	995#	995	998#	998	999#	999	1001#	1001	1002#	1002
	1005#	1005	1013#	1013	1021#	1021	,022#	1022	1023#	1023	1025#	1025	1026#	1026
	1027#	1027	1028#	1028	1029#	1029	1033#	1033	1046#	1046	1047#	1047	1048#	1048
	1049#	1049	1056#	1056	1052#	1052	1054#	1054	1056#	1056	1195#	1196#	1197#	1,98#
	1199#	1200#	1201#	1202#	1203#	1204#	.205#	1206#	1207#	1208#	1209#	1210#	1210	1215#
	1216#	12.7#	1218#	1219#	1220#	1221#	1222#	1223#	1224#	1225#	1226#	1226	1231#	1232#
	1233#	12344	1235#	1236#	1237#	1238#	,239#	1240#	1241#	1242#	1243#	1244#	1245#	1246#
	1247#	1248#	1249#	1250#	1250	1255#	.256#	1257#	1258#	1259#	1260#	1261#	1262#	1263#
	1264#	1265#	1265	1276#	1271#	1272#	1273#	1274#	1275#	1276#	1277#	1278#	1279#	1280#
	1281#	1282#	1283#	1284#	1285#	1286#	1287#	1288#	1289#	1290#	1290	1295#	1296#	1297#
	1298#	1299#	1300#	1301#	1302#	1302	1307#	1308#	1309#	1310#	1311#	1312#	1313#	1314#
	1315#	13.6#	1317#	1318#	1319#	1319	1324#	1325#	1326#	1327#	1328#	1329#	1330#	1331#
	1332#	1333#	1334#	1335#	1336#	1337#	1338#	1339#	1340#	1341#	1342#	1342	1347#	1348#
	1349#	1350#	1351#	1352#	1353#	1354#	1355#	1356#	1357#	1358#	1359#	1360#	1361#	1362#
	1363#	1364#	1365#	1366#	1367#	1368#	1368	1373#	1374#	1375#	1376#	1377#	1378#	1379#
	1380#	1381#	1382#	1383#	1384#	1385#	1386#	1387#	1388#	1389#	1390#	1391#	1391	1396#
	1397#	1398#	1399#	1400#	1401#	1402#	1403#	1404#	1405#	1406#	1407#	1408#	1409#	1410#
	1411#	1411	1416#	1417#	1418#	1419#	1420#	1421#	1422#	1423#	1424#	1425#	1426#	1427#
	1428#	1429#	1429	1435#	1436#	1437#	1438#	1439#	1440#	1441#	1442#	1443#	1444#	1445#
	1446#	1447#	1447	1452#	1453#	1454#	1455#	1456#	1457#	1458#	1459#	1460#	1461#	1462#
	1463#	1464#	1465#	1466#	1467#	1468#	1469#	1470#	1470	1475#	1476#	1477#	1478#	1479#
	1480#	1481#	1482#	1483#	1484#	1485#	1486#	1487#	1488#	1489#	1489	1494#	1495#	1496#
	1497#	1498#	1499#	1500#	1501#	1502#	1503#	1504#	1505#	1506#	1507#	1508#	1509#	1510#
	1511#	15.2#	1513#	1514#	1515#	1516#	1517#	1518#	1519#	1519	1526#	1526#	1527#	1528#
	1529#	1530#	1531#	1532#	1535#	1534#	1535#	1536#	1537#	1538#	1538	1544#	1545#	1546#
	1547#	1548#	1549#	1550#	1551#	1552#	1553#	1554#	1555#	1556#	1557#	1558#	1559#	1560#
	1561#	1562#	1563#	1564#	1565#	1566#	1566	1705#	1706#	1707#	1708#	1708	1713#	1714#
	1714	1722#	1723#	1724#	1725#	1726#	1726	3992#	3993#	3994#	3995#	3996#	3997#	3998#
	3999#	4000#	4001#	4002#	4003#	4004#	4005#	4006#	4007#	4007	4269#	4270#	4271#	4272#
	4273#	4274#	4275#	4276#	4277#	4278#	4279#	4280#	4281#	4282#	4282			
ALLIST	2105	2115#												
ALLSTK	7135	7137#												
ANDOR	4672	4752#												

AMPLOW	4716	4821#												
ARG	170	1689#												
ARGLG	170	1688#												
ARYSTR	6810#	6820												
ARYTAB	1665#	2224	5635	5727	5918	6739								
ARYVA2	6771#	6795												
ARYVAR	6744	6772#	6817											
ASC	994	7081#												
ASPA2	3933	3972#												
ASPAC	3945	3973#												
ATN	931	931#												
ATNFIX	134	931#												
ATNTB	1650#	5200												
B	880	888	1751	1755	1764	1769	1792	1794	1797	1799	1830	1832	1833	1939
	1970	1985	1991	2022	2023	2151	2159	2165	2237	2266	2305	2321	2373	2417
	2423	2437	2467	2483	2520	2531	2752	2796	2798	2829	2839	2841	2945	2951
	2954	2956	3018	3232	3235	3261	3284	3296	3336	3395	3398	3399	3403	3423
	3553	3557	3562	3565	3588	3599	3781	3973	3977	4023	4083	4087	4182	4190
	4311	4318	4424	4425	4440	4527	4610	4656	4657	4660	4693	4695	4701	4703
	4713	4714	4715	4719	4738	4758	4775	4778	4801	4821	4827	4842	4861	4863
	4864	4867	4872	4893	4910	4912	4914	4915	4927	4957	5120	5124	5213	5216
	5354	5356	5368	5384	5387	5398	5399	5449	5464	5516	5519	5533	5656	5675
	5675	5676	5677	5679	5686	5709	5711	5724	5883	5887	5949	6026	6044	6047
	6053	6054	6060	6061	6064	6073	6074	6078	6098	6118	6138	6162	6172	6189
	6194	6196	6311	6314	6331	6402	6406	6444	6472	6475	6492	6533	6535	6550
	6640	6647	6672	6674	6696	6699	6727	6771	6790	6805	6807	6808	6819	6822
	6846	6848	6856	6859	6864	6876	6888	6889	6892	6903	6905	6916	6980	6990
	6992	7025	7040	7052	7053	7063	7066	7113	7131	7135	7139	7143	7149	7152
	7155	7155	7156	7179	7196	7214	7217	7220	7227	7229	7279	7283	7297	7299
	7300	7341	7382	7373	7376	7379	7384	7387	7401	7405	7409	7411	7419	7424
	7430	7449	7460	7466	7469	7503	7507	7521	7525	7543	7544	7550		
BLTLOP	1795#	1802												
BLTU	1786#	2026	5714											
BLTUC	179C#	6898												
BOTLOW	5194#	5196												
BRKTXT	1719#	3047												
BSERR	156	5985#	6083	6152										
BUF	133	1583#	2045	2297	2481	7447	7461							
BUFLEN	115#	1583	2522											
BUFLIN	7444	7460#												
BUFMIN	1579#	2450	3853											
C	884	1749	1754	1822	2153	2290	2596	2429	2519	2528	2529	2801	2944	2952
	3337	3397	3398	3554	3597	3598	4310	4439	4691	4699	4712	4825	4865	4911
	5123	5127	5221	5224	5351	5366	5541	5559	5650	5674	5725	5943	6024	6044
	6062	6079	6163	6173	6332	6404	6537	6541	6562	6671	6804	6845	6849	6857
	6878	6880	6895	6901	6904	6978	7027	7041	7043	7128	7137	7157	7223	7225
	7374	7410	7531											
CASSW	66	1042	1127	7680										
CAT	4598	6916#												
CHEAL	2067#	2090												
CHKCOM	3917#													
CHKSTK	169#	5489	5162	6427	7015									
CHRS	946	7101#												

CHRLOW	799	2969#												
CHRGTR	2961#	2972	3651	6557										
CLEAR	1125	3207#												
CLEARC	2207#	3208	3246	3297										
CLMWID	16#	3912	3924											
CNLCA1	136	2615#												
CNLCA2	136	2631#												
CNLCA3	136	2998#												
CNLCA4	2859#													
CNLCB1	2624#													
CNLCB2	2639#													
CNTLCN	2863	3002#												
CNTNFL	162	810	1595#	1889	1945	2647	2651	3043	4044					
COLIB	2401	2406#												
COMPRT	5774	5899#												
CONIA	166#	5405#	5485											
CONINT	7107	7240	7335#											
CONSIN	166#	4975												
CONSSW	10#	1004	1129	7662										
CONT	1117	3061#												
CONTRW	14#	161	808	813	1594	1886	1942	2262	2539	2595	2642	3040	4040	
CONTW	121#	2644												
COPNUM	3486	3545#												
COS	92#	9C4#												
COSFIX	139	92A#												
CR	120#	6643												
CRDO	155	1892	2478	2604	3748	3829	3855#	3918	7433					
CRDONE	2314	2421	2449#											
CRFIN	3869#	6645												
CRUNCH	1976	2269#												
CSLOOP	5370#	5389												
CURLIN	136	1643#	1872	1920	1951	2773	2901	3022	3080	3288	3319	3369	4437	6420
CZLOOP	2079#	2082												
D	828	1756	1846	1855	1974	1979	1984	2011	2038	2041	2044	2047	2051	2067
	2087	2098	2101	2105	2109	2115	2125	2296	2347	2350	2356	2357	2367	2379
	2394	2395	2428	2430	2452	2453	2454	2455	2456	2747	2797	2801	2802	2830
	2898	2911	2921	2929	2926	3174	3188	3190	3192	3198	3200	3224	3225	3291
	3294	3353	3427	3428	3466	3477	3482	3506	3511	3520	3524	3529	3576	3711
	3729	4124	4181	4188	4207	4245	4262	4321	4362	4391	4395	4511	4513	4537
	4553	4554	4555	4566	4609	4611	4612	4662	4742	4752	4756	4779	4780	4841
	4846	4858	4894	4956	5065	5209	5212	5355	5363	5371	5373	5382	5431	5470
	5480	5489	5573	5576	5577	5582	5587	5591	5594	5628	5642	5645	5649	5654
	5655	5662	5663	5669	5737	5740	5743	5747	5880	5882	5896	5912	5921	5956
	6020	6078	6081	6116	6145	6158	6159	6267	6315	6348	6358	6371	6401	6410
	6581	6535	6546	6578	6632	6634	6761	6763	6830	6863	6869	6889	6944	6991
	6995	7039	7040	7042	7071	7165	7176	7177	7178	7298	7342	7375	7377	7407
	7445	7486	7488	7489	7520	7542	7545	7792	7793					
DADD	188#	1145												
DANDOW	4759	5484#												
DATA	1064	2785	3384#	3716	4299	6312								
DATATK	988#	2404	4329											
DATBA	4135	4158#	4335											
DATEND	4302#													

```
DATLEN    1624#   1869    432#
DATLOF    4147    42988   4331
DATPTR    1672#   2990    4095
DATSME    1868#   4021
DBLOSP    1145#   4943
DCOMP      168#   1153
DEXPRT     167#   7478#   7478
DDIV       168#   1151
DEF       1114    6307#
DEFFIN    6337    6404#
DEL       1994    7537#
DELETE    1123    7517#
DFACLO     170    1680#   4700    4887    4891
DIM       1060    5516#
DIMLEN    5497#   5517
DIMPLO    1598#   5531    5877    5910    5963    6050    6102
DIRIS     3028    3039#
DMULT      168#   1149
DNTCPY    3515    3524#
DOASIG    4201#   4217
DOCMP     4787    5396#
DOCONT    3700#   3736
DODSP     4905#   4964
DOMIN     5026    5061#
DONNOT    6183    6187    6190#
DORES     1607#   2242    2318    2409
DOSIZT    3966#
DSCIMP    16154   6508    6574    6954    7,10
DSKPLN      98    936     985     1052    1,07    7559
DSUB       168#   1147
DVBEMX     130    1677#
DVAR      6824#
DVAR2     6731    6822#
DVARS     6757    6825#
E          651    1757    1857    1874    1877    1881    1901    2039    2068    2085    2802    2896    5062    3189
          3197    3221    322c    3340    3362    3519    3597    3953    3967    4312    4319    4583    4594    4712
          5364    5370    5378    5385    5475    5738    5741    5954    5966    5985    6019    6079    6116    6143
          6264    6372    6408    6427    6603    6689    6760    6828    6892    6936    7041    7112    7197    7226
          7229    7267    7282    7300    7306    7351    7368
EATEM     5564#   5563    5569
EDIT      1097    1097#
ELSE      1091    3388#
ELSETA     994#   2383    3725    7475
END       1056    5014#
ENDCLN    4894    3019#
ENDREL    4582    4548    4566#
ENDTK      986#   2932    2936
EQU_TA    1033#   3459    6330
ERR0S     1570#   5986
ERREM     1540#   3063
ERROD     1393#   5967
ERRDIR    4073    6317    6410#
ERRDVR    1413#   1878
```

```
ERRFC     1292#   7307
ERRFIN    1916#   3050
ERRIO     1431#   6428
ERRLS     1491#   6937
ERRNR     1212#   1882
ERRO_     1267#   4313
ERRJM     15214   1858
ERRLR      147    1883    1883#   3073    3343    3365    4315    5969    5988    6375    6430    6606    6692    6939
          7309
ERRLV      136    1304#
ERRMG     1252#   3363
ERRSN     1228#   1875
ERRSU     1472#   6690
ERRST     1521#   6604
ERRTAB    1184#   1895
ERRTM      170    1489#
ERRUF     1560#   6373
ERRUS     1344#   3341
EVAL      4519    4993#   5011    6923
EXCMGI    3397#   3409
EXIGNT    4280    4288#
EXP       1408    922
EXPSTK    4667    4732#
EXTFNC      38#   139     919     928     968     1029    1049    1056    5062    5064
FAC        144    650     1684#   4983
FACOLD    4834    4927#
FACLO     1446    1681#   3500    5810    4684    4798    4870    4945    4987    5103    5166    5447    6585    6919
          6932    7018
FACSGN    4952    4970#
FADD      1418#   1155
FADLD     1438#   4408
FALSIF    3696    3711#
FBUFFR     155    1690#
FCERR      147    3095    3155    7085    7306#   7345
FCOMP     1418#   1163    4421
FDIV      1161    1181#
FIN       1418#   4213    5620    5017    7382
FINBLK    6475    7113#
FING6     5180    5213#
FINI      2015    2036#   7334
FININ     2497    3850#
FINNOW    6121    6199#
FINPTR    5658    5707#
FINWEL    4569    4773#
FINTMP    4714#   4745    4762    4805
FLGINP    1631#   4417    4498    4139    4247
FLOAT      143#
FLOATR     143#
FMULT     1159    1159#
FNDFOR    1735#   2745    5357    4381
FNDLIN    1959    2148#   5335    7523
FNDLEN    5042    6348#
FNDVAR    6786#   6908
```

FNINP	910	7239#													
FNOUT	1049	7251#													
FNTK	1021#	5040	6436												
FNWAIT	1105	7263#													
FOR	1060	2731#													
FORTR	987#	1743	2640												
FOUND	237E#														
FOUT	1414#	3602	6464												
FPWR	1408#														
FPWRQ	169#	4739#	4739												
FRCDBL	167#	1134	4880	4899											
FRCINT	167#	748	1136	3150	4754	5438	5466	7335	7766	7781					
FRCSNG	167#	1138	2790	2822	4733	4955	5205	6390							
FRCSTR	169#														
FRCTOS	1134#	3552													
FRE	900	6209#													
FREFAC	6234	6470	6626	7017#											
FRESTK	5346	7014#	7068												
FRETM2	6950	7060#													
FRETMP	3531	5359	6946	7021#	7167										
FRETOP	153	1616#	2615	6241	6668	6681	6707	6845	6895	7045	7055				
FRMLHR	2817	4511#													
FRMLVL	2785	3146	3466	3617	3783	4510#	5056	5157	6386	7331	7777				
FSUB	1157	1157#													
FUNCTS	127#	999	1621	1114	1541	4064	4071	5038	6289	6403	6413				
FUNDSP	897#	5214													
GARBAG	6234	6700#													
GARBAG	6676	6688#													
GETAGR	4076#														
GETBCD	168#	6630													
GETBYT	3096	3565	5173	7200	7277	7316	7330#	7790							
GETDEF	5986	6137#													
GETFNM	6308	6345	6435#												
GETSPA	6488	6505	6654#	7141											
GETSTR	1821#	2759	3681	4515	6633										
GETYPE	170	3792	4796	5108	5236#	6220									
GIVDBL	6225	6263#													
GIVINT	6271#														
GONE	1982	2880	2924#												
GONE2	2931#	3681													
GONE3	2929#	3705													
GOODCH	2496	2519#													
GOSUB	1086	3280#													
GOSUB	991#	3292	3361	3590											
GOTO	1074	3303#	3702												
GOTOIN	984#	3595	3655												
GRBPAS	0777	6689#													
GREATR	1036#	4540	4544												
GTBYTC	3942	7328#													
H	769	790	827	873	885	887	1735	1741	1758	1752	1753	1755	1767	1800	
	1824	1826	1837	1848	1894	1902	1922	1947	1953	2024	2028	2033	2034	2037	
	2040	2042	2050	2060	2067	2070	2075	2076	2077	2080	2086	2127	2151	2155	
	2191	2162	2164	2188	2190	2199	2243	2249	2252	2353	2377	2393	2427	2435	

	2436	2440	2449	2466	2480	2530	2783	2771	2787	2795	2819	2827	2833	2886	
	2890	2892	2897	2908	2953	2961	2988	3025	3046	3070	3148	3152	3158	3173	
	3174	3181	3186	3191	3193	3202	3215	3216	3223	3240	3285	3286	3317	3327	
	3328	3336	3338	3367	3371	3405	3484	3506	3503	3504	3520	3536	3543	3551	
	3554	3560	3567	3569	3578	3596	3661	3745	3778	3787	3788	3846	3852	3946	
	3951	3981	4025	4055	4059	4063	4079	4082	4091	4192	4203	4226	4237	4259	
	4266	4308	4320	4393	4405	4410	4415	4419	4423	4440	45.0	4606	4688	4692	
	4694	4700	4702	4707	4757	4786	4790	4800	4858	4866	4885	4889	4902	4917	
	4918	4943	4959	4970	4981	4985	5074	5086	5092	5118	5170	5203	5207	5222	
	5223	5350	5353	5365	5367	5369	5386	5443	5445	5479	5481	5488	5490	5497	
	5592	5626	5636	5644	5664	5667	5674	5708	5712	5718	5724	5729	5739	5742	
	5751	5875	5893	5908	5938	5944	5952	5953	5955	5957	6018	6025	6027	6042	
	6045	6048	6065	6065	6067	6072	6091	6100	6109	6117	6119	6137	6142	6144	
	6146	6154	6162	6172	6174	6181	6185	6202	6213	6266	6271	6331	6361	6364	
	6365	6366	6367	6368	6381	6388	6392	6393	6400	6405	6407	6409	6411	6418	
	6422	6423	6425	6483	6486	6490	6496	6507	6510	6532	6536	6539	6560	6611	
	6616	6675	6683	6709	6712	6717	6718	6747	6748	6750	6783	6789	6803	6809	
	6827	6829	6831	6842	6848	6851	6855	6856	6862	6870	6872	6875	6877	6879	
	6882	6886	6891	6892	6900	6902	6905	6906	6917	6930	6934	6952	6963	6967	
	6975	6977	6979	7024	7026	7028	7057	7087	7091	7129	7144	7145	7146	7150	
	7151	7156	7264	7333	7338	7347	7371	7373	7386	7406	7429	7441	7446	7464	
	7488	7482	7496	7508	7526	7527	7533	7550	7552	7779					
HAVTYP	5574	5594#													
IADD	1674	1105													
ICOMP	168#	1173													
IDIV	168#	1171													
IDONE	123#	2633	2861	3000											
IF	1078	3616#													
IFQNDN	2831#														
IFTK	990#	3419													
ILLFUN	194	906	7303#												
IMULT	168#	1159													
INCHK	2486	2628#	3006												
INDLGP	5882#	5981													
INEG	168#														
INIT	1414	734	1957												
INLIN	1986	2279	2480#												
INLINC	2472	2485#	2502	2507	2536										
INLINN	2474#	2512													
INLPRM	6142#	6185													
INPCOM	3484#	4209													
INPCON	4089	4696#													
INPKT	142#	1926													
INPLT	1060	4036#													
INPRMT	7263	7245#													
INRART	165#														
INT	1416														
INTSP	1165#	4859													
INTTO2	3145#	3611													
INTTUX	5144#	5685													
INTXT	155	1744#													
INXMNT	2380#	2388													
ISAMY	5619	5675#													

ISCNTL	6996#	7417												
ISFUN	5047	5120#												
ISGOSU	5592	3597#												
ISIGN	1A8#													
ISLET	3116#	5003	5535	5554	5566									
ISSEC	5550	5559#												
ISUB	160#	1167												
ISVAR	5000	5000#												
KJUMP	2294#	2415												
	830	1754	1923	2068	2153	3024	3071	3189	3220	3337	3519	3561	4439	4919
	5193	5224	5400	5442	5477	5478	5486	5487	5643	5725	6108	6183	6173	6263
	6265	6269	6332	6424	6497	6843	6849	6857	6871	6885	6893	6904	6981	6987
	6988	7152	7157	7161	7374	7551								
LABBCK	5081#	5210												
LABNUM	1054#	5128												
LEFT1	948	7123#												
LEFT2	7129#	7215												
LEFT3	7127#	7181												
LEN	938	7063#												
LEN1	7067#	7082	7365											
LENGTH	4#	22	3E	37	82	111	117	119	131	191	158	165	740	747
	789	813	874	899	901	908	933	956	970	993	995	997	999	1002
	1014	1022	1024	1029	1047	1051	1090	1099	1109	1117	1123	1133	1185	1522
	1617	1623	1645	1674	1678	1686	1691	1701	1718	1745	1748	1859	1861	1867
	1879	1897	1908	1928	1934	1992	1996	2092	2181	2183	2194	2253	2261	2327
	2380	2488	2595	2857	2731	2777	2781	2783	2788	2811	2815	2820	2855	2865
	2867	2903	2957	2960	3007	3009	3060	3084	3101	3127	3143	3205	3250	3307
	3315	3388	3411	3481	3491	3537	3539	3546	3573	3582	3619	3624	3648	3853
	3663	3685	3686	3690	3692	3694	3698	3707	3764	3790	3797	3838	3843	3937
	3938	3946	3947	3952	3990	4057	4079	4135	4142	4146	4159	4162	4167	4174
	4200	4215	4220	4225	4250	4256	4290	4301	4302	4307	4317	4318	4333	4362
	4368	4376	4385	4397	4451	4452	4472	4528	4536	4589	4592	4595	4600	4604
	4615	4655	4694	5032	5075	5079	5083	5106	5113	5188	5217	5220	5226	5246
	5332	5344	5395	5403	5410	5430	5542	5546	5548	5561	5572	5599	5610	5621
	5627	5681	5736	5745	5748	5929	5936	5950	6016	6022	6167	6191	6208	6218
	6225	6247	6254	6262	6275	6282	6325	6352	6382	6384	6445	6448	6460	6484
	6512	6587	6598	6620	6635	6749	6764	6781	6787	6791	6797	6823	6832	6883
	6975	6982	7049	707E	7088	7147	7231	7303	7311	7314	7327	7329	7339	7369
	7387	7703	7764	7769	7775	7784								
LEPSKP	1096#	1904												
LESSTK	1036#	4544												
LET	1076	2738	2934	3451#										
LINCHK	3826#													
LINGET	155	1972	2107	2.19	3173#	3304								
LINGTZ	3174#	3604												
LINLEN	114#	2601	3691	3826	3912									
LINLIN	2067#	2517												
LINPR1	142#	2916	7437											
LINPT1	2602#													
LINPT2	3495#													
LINPT3	3821#													
LINPTG	3916#													
LIST	1120	7401#												
LISTN	7406#	7456												
LISTEN	126#	130#	2854	2995	7415									
LOG	920	920#												
LOOP	2151#	2171	3330											
LOOPLN	4434	4445#												
LOOPEX	1740#	1771												
LOOPLN	3598#	3611												
LOPUT2	4111#	4240												
LOPPLA	5921#	5959												
LOPPNU	5632#	5671												
LOPPTA	6049#	6076												
LOPWEL	4539#	4564												
LPOPEN	4515#	4724	5069	5434										
LPTLEN	115#													
LPTSW	8#	163	813	91E	996	1001	1047	1108	1122	1598	1940	2256	2544	2595
	2873	3742	3812	3855	3899	3957	6276	7398						
LSTWRK	1024#	4581												
M	764	767	791	884	886	1740	1749	1751	1796	1822	2033	2039	2041	2049
	2069	2071	2079	2085	2087	2154	2156	2181	2189	2299	2322	2335	2363	2378
	2385	2418	2439	2529	2803	2875	2887	2891	2896	2898	2952	2954	2962	3112
	3402	3534	3558	3560	3587	3623	3824	3858	4080	4126	4257	4319	4321	4392
	4453	4535	4571	4611	4691	4693	4694	470.	4865	4867	4916	4918	5221	5223
	5349	5351	5354	5364	5366	5368	5385	5533	5614	5730	5738	5741	5743	5897
	5937	5942	5948	5954	5956	5978	6017	6024	6026	6047	6062	6064	6092	6116
	6118	6143	6145	6404	6406	6408	6410	6443	6482	6511	6540	6612	6706	6782
	6884	6926	6928	6830	6876	6878	6885	690.	6903	6929	6935	6976	6978	6980
	7025	7027	7043	7074	7092	7112	7130	7149	7151	7191	7219	7224	7376	7377
	7386	7387	7465	7501	7768									
MAIN	1961#	1968	2673											
MAKINT	184#	750	5483	5492	6273									
MEMSIZ	153	1611#	2E1E	3218	6701									
MIDS	932	7190#												
MID2	7292	7210#												
MINUTK	155	1028#	2114	5020										
MLOOP	7542#	7548												
MLOOPK	2047#	2054												
MORCOM	3923#	3926												
MORLIN	3177#	3204												
MORPK	3746#	3848												
MOVE	141#													
MOVFR	145#	4401												
MOVFR	142#	4096												
MOVINS	6956	6961	6973#											
MOVLP	6948#	6995												
MOVMF	145#	4413	6378											
MOVMF	1426	2793	2825	4974										
MOVMM	142#	4417	4428	6785										
MOVSTR	6499	6987#	7163											
MULDIV	34#	156	157	5754	5882	6403								
MUSICH	2359	2346#												
NEG	145#													
NEWLNK	3745#													
NEWSTT	2893#	2926	3262	3372	4437									

Tom Clancy Blanton

NEXT	1062	4351#											
NEXTC	4367#	4651											
NFERR	1881#	4387											
NMARY_	5946	5955#											
NMARY2	5940	5952#											
NMREL	4544#	4546											
NOIATI	2406	2411#											
NOUEL	2011#												
NOPKIN	2613#	2619											
NOSEC	5551	5571#											
NOTABK	3779	3421	3484	3481#									
NOTOIM	6458	6062#											
NOTER	5036	5431#											
NOTFOJ	5934	6015#											
NOTFNS	5640	5673#											
NOTFRF	5196	5208#											
NOTITI	5847	5652	5662#										
NOTLST	7050	7057#											
NOTOL	2749	2757#											
NOTWTI	4047	4063#											
NOTSTU	4535#												
NOTTHK	7272	7279#											
NUTTK	1025#	5034											
NOTTKC	2909	2922#											
NOWGET	4186	4196#											
NONLIN	4305	4327#											
NTH26	2365	2435#											
NTHIBI	2439#	2442											
NULCNT	834#	3098	3870										
NULL	1103	3085#											
NUMCHJ	2936#	2938											
NUMOFN	5120#	5130											
NUMINS	417C	42C1#											
NUMLEV	111#	1849											
NUMREL	4600#	4795											
NUMTMP	116#	119#	1614										
NUTHOS	3916#	3915											
NXTLON	2834#	4442											
NXTNES	2352#	2370											
OOONE	122#	2617											
OKGLTJ	5657	3602#											
OKNOKK	513C	5182#											
OLOLIN	1646#	3031	3077										
OLOTXT	1647#	2254	3037	3065									
OMERR	135	1857#	3038										
ONEFUN	1046#	5045	5126	5194	5200								
ONELIN	2111	2124#											
ONEON	2809	2829#											
ONGLTJ	1101	3584#											
OPRTYP	1604#	4825	4906										
OPTAB	950#	4607											
ORFIN	5475	5486#											
OUTBF_	2526	2532#											

OUTCLN	816	2536#											
OUTJC	146	807#											
OUTWRO	7254#	7322											
PARAMK	5052#	5183	6350										
PEEK	934	7765#											
PLOUP	7065#	7072											
PLUSTK	155	1027#	1629	4576	5009								
POKE	1110	7776#											
POPGCP	1760	1764#											
POPHRT	160#												
POS	913	6279#											
PPSWAT	2561	6655#											
PREAP	7124	7174	7293#										
PREAPE	7193	7297#											
PRINT	1114	3747#											
PRINTC	3751#	3984											
PRINTK	990#	2331											
PRIT3	7501#	7510											
PRIT4	7464#	7505											
PRTNL_	3872#	3884											
PS#	607	1970	1986	1987	2014	2372	2587	2588	2621	2831	3019	3045	3180	3194
	3045	3478	3525	3533	3679	3882	3937	3948	4170	4201	4394	4931	5357	5468
	5632	5888	5977	6028	6066	6149	6160	6661	6662	6685	6688	6695	6868	6861
	7212	7266	7487	7494										
PTRBET	3452	4114	4372	5089	5529#	6323								
PTRBT2	5534#	6449												
PUFOUT	167#													
PURE	/#													
PUSHF	1421#	4736	4972											
PUSHMA	871	890#	891											
PUTDEI	6407#	6515												
PUTNEW	6578#	6969	7116	7170										
PUTVAL	3563	3567#												
Q	981#	986	986#	987	987#	988	988#	989	989#	990	990#	991	991#	992
	992#	993	995#	996	996#	995	995#	998	998#	999	999#	1001	1001#	1002
	1002#	1005	1003#	1011	1011#	1012	1013	1013#	1019	1019#	1020	1021	1021#	1022
	1022#	1023	1023#	1025	1025#	1026	1026#	1027	1027#	1028	1028#	1029	1029#	1031
	1031#	1032	1032	1033#	1035	1035#	1036	1046	1046#	1047	1047#	1048	1048#	1049
	1049#	1050	1650#	1652	1052#	1054	1054#	1056	1056#	1177#	1187#	1194	1194#	1212
	1214	1214#	1220	1230	1230#	1254	1254#	1267	1269	1269#	1292	1294	1294#	
	1304	1306	1306#	1321	1323	1323#	1344	1346	1346#	1370	1372	1372#	1393	1395
	1395#	1413	1413	1415#	1431	1434	1434#	1449	1451	1451#	1472	1474	1474#	1491
	1493	1493#	1521	1524	1524#	1540	1543	1543#	1568					
QINLIN	155	2266#	4077	4153										
QINT	1414													
RAMBOT	15#													
READ	1070	4091#												
READY	135	1940#	3053	7413										
REALIO	5#	15	129	157	999	1119	2469	2474	2533	2543	2612	2628	3055	6703
REASON	135	1789	1835	1848#	6086									
RECOUY	1710#	1954	7528											
REJINP	3496#													
REM	1048	1899	3315	5592#										

TMYIN	2624#	2635												
TRYULT	2592	2611#												
TSTACK	143#													
TSTOP	4521#	6464												
TTICHN	123#	2658												
TTUCHN	124#	2623												
TTYCHN	2585#													
TTYIST	3963#													
TTYPOS	839#	2597	2609	3822	3862	3874	3910	3964	6280					
TWAM	6721#	6728												
TXTTAB	144	1655#	2149	2179	2191	2986								
UMULT	157#	6069	6156											
USERR	3340#													
USINTR	1022#													
USMLUC	900#													
VAL	94C	7362#												
VALINT	169#													
VALSNG	169#													
VALTYP	149	170	1602#	3465	4165	4587	4681	4834	4933	5237	5596	5931	6169	6591
VARTAB	1650#	2018	2030	2192	2221	3230	3916	5630	6735	7548	7554			
VINT	168#	902												
VMOVAF	168#	4683	4929											
VMOVE	169#	3541	6594											
VMOVFA	166#													
VMOVFM	167#	5111												
VMOVMF	167#	3571												
VNEG	167#	5077												
VPUSHC	4697	4705	4712#											
VSIGN	166#	3688#	3680											
ZERITA	6091#	6096												
ZERO	1418	7366												
ZERUER	5729#	5734												
SCOUT	84	84#	676	1736	1765	1849	1948	2099	2116	2258	2799	3175	3182	3233
	4363	4781	6455	6139	6214	6710								
$START	84#	84												
.CI	133	7860#												
.P	110#	735	736#	749	750#	751	752#	774	775#	800	801#	811	812#	817
	818#	831	852#	855	856#	872	873#	877	878#	892	893#	898	899#	903
	904#	905	906#	907	908#	909	910#	911	912#	914	915#	916	917#	918
	919#	921	922#	923	924#	925	926#	927	928#	930	931#	932	933#	935
	936#	939	940#	941	942#	943	944#	945	946#	947	948#	949	950#	951
	952#	953	954#	1059	1060#	1061	1062#	1063	1064#	1065	1066#	1067	1068#	1069
	1070#	1071	1072#	1073	1074#	1075	1076#	1077	1078#	1079	1080#	1081	1082#	1083
	1084#	1085	1086#	1087	1088#	1089	1090#	1092	1093#	1094	1095#	1096	1097#	1098
	1099#	1100	1101#	1102	1103#	1104	1105#	1106	1107#	1111	1112#	1113	1114#	1115
	1116#	1118	1119#	1121	1,22#	1124	1125#	1126	1127#	1131	1132#	1135	1136#	1137
	1138#	1139	1140#	1146	1,47#	1148	1149#	1150	1151#	1152	1153#	1154	1155#	1156
	1157#	1158	1159#	1160	1,61#	1162	1163#	1164	1165#	1166	1167#	1168	1169#	1170
	1171#	1172	1173#	1174	1,75#	1737	1738#	1761	1762#	1766	1767#	1772	1773#	1790
	1791#	1803	1804#	1829	1830#	1836	1837#	1850	1851#	1864	1865#	1870	1871#	1873
	1874#	1885	1886#	1890	1891#	1893	1894#	1896	1897#	1900	1901#	1905	1906#	1918
	1919#	1921	1922#	1927	1928#	1946	1947#	1949	1950#	1952	1953#	1955	1956#	1958
	1959#	1963	1964#	1969	1970#	1973	1974#	1977	1978#	1983	1984#	1990	1991#	1995

	1996#	2016	2017#	2019	2020#	2027	2028#	2031	2032#	2046	2047#	2055	2056#	2058
	2059#	2074	2075#	2083	2084#	2091	2092#	2100	2101#	2104	2105#	2108	2109#	2112
	2113#	2117	2118#	2120	2121#	2123	2124#	2150	2151#	2172	2173#	2180	2181#	2186
	2187#	2193	2194#	2198	2199#	2209	2210#	2213	2214#	2216	2217#	2219	2220#	2222
	2225#	2225	2226#	2228	2229#	2240	2241#	2245	2246#	2248	2249#	2251	2252#	2255
	2256#	2260	2261#	2265	2266#	2280	2281#	2295	2296#	2298	2299#	2304	2305#	2311
	2312#	2315	2316#	2319	2320#	2325	2326#	2334	2335#	2342	2341#	2345	2346#	2352
	2353#	2362	2363#	2366	2367#	2371	2372#	2389	2390#	2402	2403#	2407	2408#	2418
	2419#	2416	2417#	2422	2423#	2426	2427#	2433	2434#	2443	2444#	2448	2449#	2451
	2452#	2473	2474#	2479	2480#	2482	2483#	2487	2488#	2493	2494#	2498	2499#	2503
	2504#	2508	2509#	2513	2514#	2518	2519#	2527	2528#	2537	2538#	2542	2543#	2593
	2594#	2598	2599#	2605	2606#	2610	2611#	2620	2621#	2636	2637#	2648	2649#	2652
	2653#	2736	2737#	2739	2740#	2746	2747#	2750	2751#	2760	2761#	2766	2767#	2774
	2775#	2786	2787#	2791	2792#	2794	2795#	2800	2801#	2810	2811#	2818	2819#	2823
	2824#	2826	2827#	2830	2831#	2864	2865#	2870	2871#	2881	2882#	2885	2886#	2895
	2896#	2902	2903#	2906	2907#	2910	2911#	2917	2918#	2927	2928#	2935	2936#	2941
	2942#	2950	2951#	2973	2974#	2987	2988#	2991	2992#	3004	3005#	3017	3018#	3023
	3024#	3029	3030#	3032	3033#	3035	3036#	3038	3039#	3044	3045#	3048	3049#	3051
	3052#	3054	3055#	3066	3067#	3074	3075#	3078	3079#	3081	3082#	3087	3088#	3096
	3097#	3099	3100#	3102	3103#	3147	3148#	3151	3152#	3156	3157#	3176	3177#	3183
	3184#	3187	3188#	3205	3206#	3209	3210#	3212	3213#	3219	3220#	3228	3229#	3231
	3232#	3234	3235#	3239	3240#	3243	3244#	3247	3248#	3253	3254#	3258	3259#	3263
	3264#	3266	3267#	3282	3283#	3289	3290#	3305	3306#	3316	35,7#	3320	3321#	3331
	3332#	3334	3335#	3344	3345#	3350	3359#	3366	3367#	3370	3371#	3373	3374#	3410
	3411#	3422	3423#	3431	3432#	3453	3454#	3464	3465#	3469	3470#	3475	3476#	3487
	3488#	3498	3491#	3501	3502#	3509	3510#	3514	3515#	3517	3518#	3523	3524#	3532
	3533#	3542	3543#	3553	3554#	3564	3565#	3572	3573#	3586	3587#	3593	3594#	3602
	3603#	3605	3606#	3612	3613#	3618	3619#	3652	3653#	3658	3659#	3689	3690#	3697
	3698#	3703	3704#	3706	3707#	3717	3718#	3726	3729#	3733	3734#	3737	3738#	3749
	3750#	3763	3764#	3769	3770#	3775	3776#	3780	3781#	3784	3785#	3793	3794#	3796
	3797#	3803	3804#	3808	3809#	3811	3812#	3823	3824#	3830	3831#	3833	3834#	3842
	3843#	3849	3850#	3854	3855#	3863	3864#	3871	3872#	3875	3876#	3885	3886#	3911
	3912#	3919	3920#	3922	3923#	3927	3928#	3934	3935#	3943	3944#	3956	3957#	3965
	3966#	3970	3971#	3980	3981#	3985	3986#	4016	4019#	4022	4023#	4027	4028#	4038
	4031#	4033	4034#	4045	4046#	4048	4049#	4052	4053#	4058	4059#	4074	4075#	4078
	4077#	4086	4087#	4090	4091#	4094	4095#	4099	4100#	4115	4116#	4134	4135#	4140
	4141#	4148	4149#	4154	4155#	4166	4167#	4173	4174#	4187	4188#	4198	4199#	4205
	4206#	4210	4211#	4214	4215#	4218	4219#	4230	4231#	4235	4236#	4201	4242#	4248
	4249#	4255	4256#	4261	4262#	4265	4266#	4300	4301#	4306	4307#	4316	4317#	4325
	4326#	4332	4333#	4336	4337#	4364	4365#	4373	4374#	4377	4378#	4382	4383#	4388
	4389#	4402	4403#	4409	4410#	4414	4415#	4418	4419#	4422	4423#	4429	4430#	4433
	4434#	4438	4439#	4443	4444#	4450	4451#	4456	4459#	4462	4463#	4516	4517#	4520
	4521#	4523	4524#	4526	4527#	4543	4544#	4549	4550#	4558	4559#	4561	4562#	4565
	4566#	4571	4571#	4574	4575#	4588	4589#	4599	4600#	4608	4609#	4659	4660#	4668
	4669#	4673	4674#	4686	4683#	4687	4688#	4690	4691#	4698	4699#	4706	4707#	4709
	4710#	4717	4718#	4722	4723#	4725	4726#	4734	4735#	4737	4738#	4740	4741#	4746
	4747#	4755	4756#	4760	4761#	4763	4764#	4782	4783#	4788	4789#	4793	4794#	4796
	4797#	4799	4800#	4803	4804#	4806	4807#	4826	4827#	4832	4833#	4835	4836#	4840
	4841#	4847	4848#	4853	4854#	4857	4858#	4860	4861#	4871	4872#	4881	4882#	4884
	4885#	4888	4889#	4892	4893#	4897	4898#	4900	4901#	4904	4905#	4927	4928#	4930
	4931#	4934	4935#	4941	4942#	4946	4947#	4949	4950#	4956	4957#	4961	4962#	4965
	4966#	4973	4974#	4976	4977#	4980	4981#	4984	4985#	4988	4989#	4991	4992#	5001
	5002#	5004	5005#	5007	5008#	5012	5013#	5018	5019#	5023	5024#	5029	5030#	5037

5036#	5043	5044#	5046	5049#	5057	5058#	5070	5071#	5073	5074#	5078	5079#	5090	
5091#	5104	5105#	5109	5110#	5112	5113#	5133	5134#	5150	5159#	5163	5164#	5167	
5168#	5170	5175#	5181	5182#	5184	5185#	5199	5200#	5206	5207#	5211	5212#	5215	
5218#	5238	5239#	5347	5348#	5360	5361#	5390	5391#	5394	5395#	5406	5407#	5409	
5410#	5435	5436#	5439	5440#	5448	5449#	5455	5456#	5467	5468#	5476	5477#	5484	
5485#	5493	5494#	5515	5519#	5532	5535#	5536	5537#	5539	5540#	5551	5552#	5555	
5556#	5558	5559#	5564	5565#	5567	5568#	5570	5571#	5575	5576#	5597	5598#	5613	
5614#	5620	5621#	5625	5626#	5631	5632#	5636	5637#	5641	5642#	5648	5649#	5653	
5654#	5659	5660#	5674	5675#	5707	5708#	5715	5716#	5721	5722#	5728	5729#	5735	
5736#	5876	5874#	5886	5887#	5902	5903#	5907	5908#	5911	5912#	5919	5920#	5926	
5927#	5932	5933#	5935	5936#	5941	5942#	5947	5948#	5960	5961#	5964	5965#	5970	
5971#	5983	5984#	5989	5990#	6031	6032#	6034	6035#	6040	6041#	6051	6052#	6056	
6057#	6059	6060#	6070	6071#	6077	6078#	6084	6085#	6087	6088#	6090	6091#	6097	
6098#	6103	6104#	6107	6108#	6115	6116#	6122	6123#	6140	6141#	6153	6154#	6157	
6158#	6166	6167#	6170	6171#	6180	6181#	6184	6185#	6188	6189#	6201	6202#	6211	
6212#	6215	6216#	6221	6222#	6224	6225#	6231	6232#	6235	6236#	6238	6239#	6242	
6243#	6274	6275#	6281	6282#	6309	6310#	6311	6314#	6318	6319#	6324	6325#	6338	
6339#	6346	6347#	6351	6352#	6376	6377#	6379	6380#	6387	6388#	6391	6392#	6397	
6398#	6421	6422#	6431	6432#	6441	6442#	6450	6451#	6465	6466#	6468	6469#	6471	
6472#	6474	6475#	6489	6490#	6495	6496#	6500	6501#	6506	6507#	6509	6510#	6516	
6517#	6545	6546#	6549	6550#	6553	6554#	6558	6559#	6565	6566#	6570	6571#	6580	
6581#	6583	6584#	6586	6587#	6592	6593#	6595	6596#	6607	6608#	6610	6611#	6621	
6622#	6627	6628#	6631	6632#	6646	6647#	6650	6651#	6665	6666#	6669	6670#	6679	
6680#	6682	6683#	6693	6694#	6698	6699#	6702	6703#	6708	6709#	6711	6712#	6715	
6716#	6720	6721#	6724	6725#	6729	6730#	6732	6733#	6736	6737#	6740	6741#	6745	
6746#	6755	6756#	6758	6759#	6769	6770#	6775	6776#	6780	6781#	6786	6787#	6796	
6797#	6802	6803#	6813	6814#	6818	6819#	6821	6822#	6826	6827#	6896	6897#	6899	
6900#	6909	6910#	6920	6921#	6924	6925#	6928	6929#	6933	6934#	6940	6941#	6943	
6944#	6947	6948#	6951	6952#	6955	6956#	6959	6960#	6962	6963#	6965	6966#	6970	
6971#	6996	6997#	7016	7017#	7019	7020#	7023	7024#	7037	7038#	7046	7047#	7051	
7052#	7056	7057#	7065	7066#	7069	7070#	7083	7084#	7086	7087#	7095	7096#	7105	
7106#	7108	7109#	7111	7112#	7117	7118#	7125	7126#	7134	7135#	7142	7143#	7160	
7161#	7164	7165#	7166	7169#	7171	7172#	7175	7176#	7182	7183#	7194	7195#	7203	
7204#	7209	7210#	7210	7217#	7241	7242#	7244	7245#	7249	7250#	7253	7254#	7265	
7266#	7273	7274#	7278	7279#	7286	7287#	7310	7311#	7317	7318#	7320	7321#	7323	
7324#	7332	7333#	7336	7337#	7346	7347#	7364	7365#	7367	7368#	7383	7384#	7404	
7405#	7414	7415#	7418	7419#	7427	7428#	7434	7435#	7438	7439#	7445	7446#	7448	
7449#	7453	7454#	7456	7457#	7459	7460#	7462	7463#	7473	7474#	7479	7480#	7485	
7486#	7493	7494#	7499	7500#	7506	7507#	7511	7512#	7519	7520#	7524	7525#	7529	
7530#	7532	7533#	7535	7536#	7541	7542#	7549	7550#	7555	7556#	7767	7768#	7774	
7775#	7778	7779#	7782	7783#	7791	7792#	7800							

ACALF	51#	1710	1715	1719	4008	4283								
AOC	3427	5397												
AOJ	5824	5967	5614	6935										
AOJ	5400													
AOM	748	750	897	902	904	906	908	910	913	915	917	920	922	924
	920	929	931	934	938	940	942	944	946	948	950	952	1058	1060
	1062	1064	1066	1068	1070	1072	1074	1076	1078	1080	1082	1084	1086	1088
	1091	1093	1095	1097	1099	1101	1103	1105	1110	1112	1114	1117	1120	1123
	1125	1130	1134	1136	1138	1145	1147	1149	1151	1153	1155	1157	1159	1161
	1163	1165	1167	1169	1171	1173								
AOMP	955#	957#	961	963	965	967	969	972	974					
ANA	1925	3025	5399	5477	5480	7283								
ANI	2350	2616	2834	2840	2864	2999	3547							
CALL	1780	1834	1883	1891	1898	1916	1956	1961	1971	1975	1988	2025	2056	2106
	2184	2217	2477	2485	2737	2744	2756	2764	2784	2789	2792	2816	2821	2824
	2915	3002	3085	3145	3149	3210	3256	3287	3303	3314	3356	3451	3467	3488
	3530	3540	3570	3584	3603	3616	3687	3715	3782	3791	3821	3826	3831	3941
	4028	4050	4056	4072	4076	4113	4152	4196	4212	4298	4380	4400	4407	4412
	4416	4420	4427	4460	4514	4518	4732	4735	4753	4791	4879	4882	4895	4898
	4920	4954	4971	4974	4978	5002	5055	5066	5076	5088	5107	5156	5161	5172
	5182	5345	5358	5484	5435	5437	5485	5534	5553	5565	5713	5884	6032	6068
	6085	6155	6219	6229	6233	6307	6316	6322	6344	6349	6377	6385	6389	6463
	6480	6489	6487	6493	6498	6524	6563	6593	6619	6625	6629	6756	6784	6897
	6922	6928	6941	6945	6949	6957	6960	7014	7067	7081	7103	7106	7123	7140
	7158	7182	7166	7173	7192	7207	7239	7251	7263	7276	7315	7330	7334	7362
	7381	7402	7416	7432	7436	7443	7451	7454	7517	7522	7530	7765	7776	7780
	7784													
CC	1993	3329	3521	5284										
CMPBET	448	1964	1986	2355	2924	3144	3177	3214	3700	3723	3746	3982	4100	4211
	4227	4258	4527	4459	4562	4998	5125	5498	5545	5560	5598	6203	6394	7270
	7328	7350												
CMA	2649	3928	3966	5375	5441	5444	6670							
CMC	2166	2168	2977	3118	5391									
CMP	767	2079	2363	2423	3403	3423	4554	4612	5378	5385	5644	5650	5656	5937
	5945	5949	5978	6546	6550	6694	7131	7219	7226					
CNC	5332	3828	5917	6568										
CNZ	1925	2118	4263	4371	5110									
COMPAR	468	1763	1795	1853	2163	3184	3236	3321	3510	3518	3637	3732	3928	6094
	6134	6369	6567	6800	6676	6726	6742	6777	6815	6847	6853	7831	7848	7423
	7546													
CMI	192	1742	2300	2307	2326	2336	234,	2382	2403	2489	2494	2499	2504	2509
	2514	2521	2589	2599	2643	2804	2877	2937	2963	2969	2974	3005	3090	3113
	3110	3362	3408	3482	3589	3606	3648	3654	3724	3759	3765	3771	3776	3825
	3914	3949	4038	4130	4158	4183	423,	4328	4454	4545	4550	4580	4590	4663
	4669	4683	4774	4826	4836	4843	4849	4936	5008	5013	5019	5025	5033	5039
	5124	5195	5201	5239	5471	5579	5584	5588	5615	5898	6554	6642	6751	6792
	7199	7474												
CZ	2387	2803	2862	3650	3840	6596	6644	7477						
DAD	1739	1769	1832	1833	1852	2023	2752	2951	3190	3191	3192	3193	3200	3235
	3557	4649	4863	4864	4914	5216	5669	5704	5921	6081	6109	6110	6159	6174
	6181	6185	6189	6196	6216	6676	6763	6790	6807	6808	6886	7053	7155	7375
DE	986	987	988	989	990	991	992	993	994	995	998	999	1001	1002
	1005	1013	1021	1022	1025	1025	1026	1027	1028	1029	1033	1046	1047	1048

	1044	1050	1052	1054	1056	1.94	1214	1230	1254	1269	1294	1306	1323	1346
	1376	1395	1415	1434	1451	1474	1493	1524	1543	1704	1712	1721	1991	4268
JCE	1174#	1188#	1195	1215	1229	1253	1268	1293	1305	1322	1345	1371	1394	1414
	1435	1450	1473	1492	1525	1542								
JCI	982#	986	987	988	989	990	991	992	993	994	995	998	999	1001
	1002	1005	1013	1021	1022	1023	1025	1026	1027	1028	1029	1033	1046	1047
	1048	1049	1050	1052	1054	1056								
JCC	1176#	1180#	1194	1214	1230	1254	1269	1294	1306	1323	1346	1372	1395	1415
	1434	1451	1474	1493	1524	1543								
JCR	1990	1966	2467	2974	3598	3729	3872	3977	5241	5242	5243	5382	5383	6074
	6161	6654	6988	7218	7302	7495								
JCX	1799	1800	2157	2199	2468	2988	3173	3213	3338	3596	3661	3745	3787	3946
	4082	4192	4226	4237	4516	5497	5592	5729	6091	6202	6364	6365	6366	6367
	6393	6532	6875	6877	6882	6891	6906	7024	7026	7028	7042	7269	7347	
JI	732													
PSIGN	478	2828												
IN	2813	2829	2837	2856	2996	7245	7280							
INR	1928	1965	2396	2429	2437	2551	2606	2978	3026	3089	3972	5379	5396	5587
	5591	5695	6541	6632	6987	7301								
INX	769	790	885	887	1741	1750	1752	1824	1902	2034	2037	2040	2042	2050
	2051	2060	2070	2075	2076	2077	2080	2086	2155	2188	2190	2356	2393	2395
	2427	2430	2440	2453	2455	2530	2832	2842	2886	2898	2892	2897	2953	2961
	3293	3328	3405	3503	3504	3559	4079	4308	4328	4393	4692	4694	4700	4702
	4868	4917	5226	5330	5353	5365	5367	5386	5387	5445	5654	5662	5663	5677
	5674	5680	5759	5762	5767	5938	5944	5952	5953	5955	5957	6018	6025	6027
	6042	6043	6048	6061	6063	6065	6098	6117	6119	6137	6144	6146	6405	6407
	6409	6422	6483	6539	6560	6618	6647	6675	6683	6747	6748	6752	6783	6829
	6827	6829	6831	6932	6977	6979	6992	6993	7087	7146	7150	7371	7466	7468
	7489	7507	7520	7544	7545	7552								
JC	2330	2343	2501	2591	2933	3185	3226	3781	4541	4556	4671	4999	5131	5197
	5531	5549	5556	5562	6082	6176	6677	6938	7132	7425				
JM	3154	4696	7304											
JMP	732	748	835	875	890	1770	1801	1862	2089	2170	2278	2431	2446	2535
	3052	3203	3245	3264	3342	3429	3610	3704	3735	3847	3883	3932	3983	4088
	4206	4216	4334	4441	4563	4723	4744	4761	4804	4947	4963	4989	5179	5392
	5407	5453	5482	5491	5670	5987	6272	6336	6429	6448	6514	6648	6767	6907
	6965	6994	7093	7115	7169	7180	7247	7308	7457	7509	7772			
JNC	1981	2506	2525	2939	3094	3237	3512	3920	3925	3968	4547	4831	5205	5846
	5560	6151	7547											
JNZ	776	853	1923	2253	2081	2121	2523	2364	2405	2414	2471	2540	2618	2634
	2708	2808	2883	3049	3364	3422	3485	3726	3731	3978	4020	4046	4146	4171
	4233	4239	4255	4304	4330	4386	4456	4568	4794	5646	5651	5733	5939	5945
	5956	5968	6075	6095	6120	6164	6222	6395	6551	6730	6753	6794	7049	7344
	7497													
JP	2368	2441	7471	7491										
JPO	4704	6186												
JZ	1759	1967	2014	2072	2102	2110	2302	2309	2313	2332	2360	2400	2420	2426
	2491	2496	2511	2516	2879	2893	2908	2971	3027	3072	3207	3251	3408	3591
	3600	3656	3695	3747	3761	3767	3773	3778	3794	3954	4084	4132	4185	4228
	4314	4431	4597	4665	4685	4830	4838	4845	4855	4939	5010	5016	5021	5027
	5035	5041	5388	5474	5618	5639	5657	5906	5933	5981	6057	6182	6374	6543
	6547	6605	6691	6743	6778	6816	7084	7201	7271	7284	7365	7412		
LIA	809	849	2317	2596	2646	2904	3462	3821	3869	3909	3963	4016	4138	4164

	4246	4586	4660	4833	4905	5236	5611	5930	5962	6049	6101	6105	6168	6279
LDAX	2041	2357	2367	3524	5384	5642	5649	5655	6648	6990	7178	7488	7542	
LHLD	1821	1868	1919	2017	2148	2178	2196	2211	2220	2238	2256	2772	2834	2985
	3021	3033	3084	3676	3217	3229	3287	3318	3499	3507	3515	3809	4031	4092
	4440	4524	4720	4797	4869	5071	5165	5629	5634	5705	5876	5917	5924	6113
	6194	6249	6236	6240	6419	6581	6663	6667	6700	6713	6722	6734	6738	6773
	6811	6844	6894	6918	6931	6953	7017	7021	7044	7109	7539			
LXI	1735	1764	1848	1894	1947	1953	2044	2096	2115	2243	2249	2296	2350	2449
	2480	2798	2925	2948	3046	3174	3181	3232	3261	3371	3551	3562	3852	4025
	4203	4259	4562	4606	4657	4688	4707	4715	4738	4756	4780	4786	4801	4858
	4902	4959	5209	5213	5516	5573	6054	6138	6213	6311	6472	6507	6578	6696
	6769	6718	6721	6819	6963	7063	7214	7446	7460	7483	7527	7533		
MCSSIM	84													
MOV	764	791	827	830	884	886	1740	1749	1751	1754	1755	1756	1796	1822
	1922	1978	2033	2039	2041	2049	2087	2068	2069	2085	2087	2151	2153	2154
	2187	2189	2299	2305	2321	2322	2335	2373	2378	2417	2418	2519	2520	2528
	2529	2881	2882	2883	2875	2887	2896	2898	2944	2952	2954	2962	3024	3070
	3116	3152	3188	3189	3197	3220	3222	3223	3225	3336	3337	3347	3398	3399
	3400	3428	3519	3520	3534	3554	3556	3560	3561	3587	3588	3597	3599	3623
	3955	3973	4080	4126	4101	4102	4310	4319	4321	4392	4439	4440	4453	4535
	4555	4586	4571	4585	4610	4611	4662	4691	4693	4699	4701	4712	4713	4773
	4823	4827	4841	4842	4846	4865	4867	4911	4916	4918	4919	5123	5127	5193
	5221	5223	5224	5349	5351	5354	5364	5366	5388	5370	5373	5440	5442	5443
	5445	5475	5478	5479	5481	5487	5488	5490	5533	5541	5559	5594	5628	5643
	5664	5674	5675	5724	5725	5738	5741	5743	5896	5897	5942	5948	5954	5956
	6017	6019	6024	6026	6044	6047	6053	6062	6064	6078	6079	6100	6108	6116
	6118	6143	6145	6162	6165	6172	6173	6263	6265	6266	6269	6271	6331	6332
	6404	6406	6408	6410	6423	6444	6482	6497	6511	6535	6540	6562	6612	6671
	6740	6782	6784	6804	6828	6830	6842	6843	6848	6849	6856	6857	6871	6876
	6878	6885	6889	6896	6892	6893	6901	6903	6904	6905	6929	6976	6978	6980
	6981	7025	7027	7040	7041	7043	7052	7071	7074	7092	7112	7128	7130	7135
	7144	7151	7156	7156	7157	7161	7191	7223	7224	7227	7229	7300	7342	7351
	7360	7373	7374	7376	7377	7380	7387	7409	7465	7501	7550	7551	7768	
MOVMI	608													
MVI	1830	1857	1874	1877	1881	2269	2272	2294	2330	2385	2485	2523	2732	2806
	2839	2912	2918	2945	3062	3178	3291	3340	3353	3362	3395	3555	3711	3834
	3850	3859	3866	3974	4041	4149	4186	4190	4312	4511	4537	4742	4861	4912
	5065	5120	5431	5577	5582	5667	5730	5880	5966	5985	6020	6092	6372	6427
	6437	6533	6537	6580	6603	6672	6689	6761	6805	6886	6936	7101	7137	7153
	7197	7260	7267	7300	7439	7449								
NOP	882	883												
ORA	812	852	1757	2052	2071	2156	2312	2328	2368	2419	2439	2882	2891	2907
	3071	3153	3401	3721	3839	4019	4081	4141	4289	4257	4303	4311	4567	4594
	5371	5374	5486	5489	5965	6052	6104	6424	6443	6542	6659	6826	6872	7075
	7363	7410	7467	7490	7502									
ORI	2374													
OUT	2822	7254												
OUTCHR	458	2271	2274	2470	2475	2534	2914	2920	3836	3865	3868	3881	3976	4151
	6641	7442												
PCHL	5566	4920	5225	6835										
POP	875	1767	1.94	1857	1858	1939	1979	1984	2011	2012	2022	2028	2038	2105
	2125	2162	2164	2165	2237	2372	2377	2379	2435	2587	2621	2747	2795	2827
	2921	3018	3045	3156	3194	3202	3244	3284	3327	3367	3477	3478	3506	3511

	3533	3536	3543	3567	3570	3578	3781	3846	3882	3948	3981	4023	4059	4083
	4201	4245	4266	4318	4410	4415	4423	4424	4527	4756	4821	4868	4885	4889
	4892	4894	4915	4931	4943	4957	4958	4976	4981	4985	5086	5118	5207	5355
	5363	5369	5398	5449	5468	5470	5638	5711	5716	5737	5740	5751	5887	5888
	5908	5977	6028	6050	6072	6073	6142	6158	6160	6194	6358	6361	6381	6392
	6400	6401	6402	6411	6425	6490	6492	6501	6611	6661	6685	6688	6771	6803
	6851	6859	6860	6861	6869	6870	6900	6944	6973	7057	7091	7113	7143	7144
	7165	7176	7212	7279	7297	7298	7338	7364	7386	7401	7406	7407	7411	7424
	7441	7464	7494	7496	7520	7525	7792							
PUSH	57#	4693	4957											
PUSH	807	888	1755	1792	1826	1846	1970	1974	1984	1985	1987	1991	2024	2101
	2104	2127	2159	2252	2266	2347	2353	2436	2588	2763	2771	2787	2796	2797
	2819	2829	2830	2831	2833	2841	2911	2928						

2455	5119	5148	5174	5180	5218									
	5283	5286	5294	5295	5317	3465	3466	3484	3484	3502	3525	3565	3569	3770
	5780	5879	5937	5951	4035	4063	4087	4091	4124	4170	4207	4262	4391	4394
	4395	4435	4419	4513	4656	4660	4695	4703	4714	4719	4752	4757	4778	4779
	4760	4866	4872	4910	4927	5074	5092	5124	5170	5203	5212	5356	5357	5464
	5519	5576	5626	5632	5673	5676	5708	5712	5875	5882	5883	5893	5912	6066
	6067	6149	6154	6314	6315	6348	6368	6371	6386	6418	6475	6486	6496	6510
	6550	6662	6695	6699	6712	6717	6789	6822	6855	6862	6863	6864	6879	6916
	6917	6930	6934	6952	6967	7039	7066	7129	7139	7145	7177	7196	7217	7266
	7249	7353	7379	7405	7419	7429	7430	7482	7486	7487	7521	7526	7779	
PLSHRM		496	6362											
PUSHR		488	2160	2161	3505	4304	4710	4711	6356	6359	6362	6363	6491	7090	7372
		7400	7428											
PUSHR		548												
RAL		4552												
RC		3115	3339	4577	6850	7228								
RET		852	858	860	1038	2267	2457	2625	2653	2980	2993	3003	3100	3107	3119
		3159	3544	3579	4034	4267	4873	5004	5087	5119	5244	5593	5752	6204	6412
		6506	6613	6686	6865	7058	7076	7230	7256	7298	7352	7393	7556	7794	
RLC		2943	4408	5122											
RNC		797	1858	2169	2965	5178	4582	4615	4776	5380	6858	7222			
RNZ		829	1744	2177	2645	3021	3010	3014	3261	3288	3215	3352	3608	6426	7034
		7304													
RST		1763	1795	1853	1964	1986	2115	2160	2161	2163	2271	2274	2355	2470	2475
		2534	2779	2828	2914	2920	2924	3144	3177	3184	3214	3236	3321	3458	3505
		3516	3518	3554	3659	3700	3723	3746	3836	3865	3868	3881	3944	3976	3982
		4093	4111	4151	4156	4211	4227	4238	4309	4327	4459	4562	4710	4711	4998
		5056	5058	5125	5155	5159	5498	5500	5545	5560	5598	5637	5732	5903	5928
		6094	6150	6203	6319	6327	6329	6356	6359	6362	6363	6369	6394	6435	6491
		6567	6600	6641	6676	6726	6742	6777	6815	6847	6853	7031	7048	7090	7205
		7210	7270	7274	7294	7324	7328	7350	7372	7408	7420	7423	7442	7546	7707
RZ		1766	1798	2158	2167	2929	3402	3404	3722	3751	3876	5372	5376	5499	5581
		5580	5590	6039	6041	6073	6989	7470							
SBH		3224	5402	6067											
SBUB		870	1071	1950	2029	2191	2207	2214	2223	2226	2246	2253	2868	2980	2989
		3015	3036	3636	3079	3241	3368	3473	4323	4375	4436	4521	4559	4572	4886
		4892	4944	4982	4986	5102	5446	5719	5726	5905	5909	6038	6088	6584	6608
		6680	6706	6880	7035	7054	7553								
SPHL		2241	2755	3559	4389	4465									
STA		1886	1944	2063	2291	2408	2808	2856	2734	3042	3097	3124	3861	3873	4043
		4497	4624	4936	5530	5595	5823	6029	6439	6590	7242	7318	7321		

STAX	1797	2390	2428	2456	2454	2456	3529	6991	7469	7503	7543	7793		
SUB	826	831	3221	4425	6284	7179	7225							
SUI	2398	2411	2931	3195	3418	3923	4539	4575	5044	6175	7484			
SYNCHK	42#	2113	2179	3458	3594	3659	3944	4053	4111	5052	5058	5153	5159	5500
	5903	6319	6327	6329	6435	7205	7210	7274	7294	7324	7787			
XCHG	1756	1792	1847	1854	2032	2084	2088	2124	2381	2392	2438	2444	2757	2899
	2923	2947	2956	2984	2992	3075	3082	3157	3201	3240	3535	4202	4252	4322
	4326	4435	5698	5104	5171	5175	5633	5746	5889	5894	5923	5927	6071	6880
	6116	6198	6212	6239	6561	6666	6684	6721	6725	6737	6741	6772	6776	6810
	6814	6956	7020	7032	7190	7293	7337	7422	7431	7537				
XRA	1887	1943	2076	2290	3041	3103	3527	3880	4096	4553	5377	5529	5540	5622
	6270	6759	6825	7470	7.26	7282								
XTHL	765	776	889	1793	1821	1825	2020	2126	2743	2768	2775	2837	3290	3374
	3471	4108	4117	4220	4236	4403	5166	5177	5187	5879	5891	6147	6333	6354
	6559	6652	6854	6921	6925	6948	6966	6974	7121	7213	7378	7421	7428	7536
	7783													

```
SPOOLER RUNTIME 48 SECONDS, 310 ACS, 1637 DISK READS, 123 DISK WRITES, 386 PAGES
 **END** USER MITS  [6000,6000] JOB  F3 SEQ. 42 DATE 10-SEP-75 09:22:31 MONITOR ALBUQUERQUE SCHOOLS 5078 **END**
```

```
LPTSPL VERSION 6(344)  RUNNING ON LPT8
 *START* USER MTA  (6000,6000) JOB  14 SEQ. 205 DATE 27-AUG-75 07:23:23 MONITOR ALBUQUERQUE SCHOOLS 507A *START*
REQUEST CREATED: 27-AUG-75  00:11:25
FILE: DSK0:F4(6000,6000) CREATED: 27-AUG-75 06:10:00 PRINTED: 27-AUG-75 07:23:30
QUEUE SWITCHES:  /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:400 /FORMS:NORMAL
FILE WILL BE DELETED AFTER PRINTING



/MEDIA/
/ON MACHINE/
/CASSETTE/
```

2454	3174	3178	3179	3180	3216									
	3285	3286	3294	3296	3317	3465	3466	3480	3484	3502	3525	3565	3569	3770
	3780	3879	3937	3951	4035	4083	4087	4091	4124	4170	4207	4262	4391	4394
	4395	4405	4414	4513	4656	4660	4695	4703	4714	4719	4752	4757	4778	4779
	4790	4870	4876	4916	4921	5074	5092	5124	5170	5203	5212	5256	5357	5464
	5514	5578	5626	5632	5673	5676	5708	5712	5875	5882	5883	5895	5912	6006
	6267	6149	6154	6314	6315	6348	6366	6371	6388	6418	6475	6486	6496	6510
	6536	6592	6693	6699	6712	6717	6729	6822	6855	6862	6863	6864	6879	6916
	6917	6930	6934	6952	6987	7039	7066	7129	7139	7145	7171	7196	7217	7266
	7299	7433	7579	7485	7419	7429	7434	7482	7486	7487	7521	7526	7779	
PUSHM	484	6382												
PUSHM	488	2184	2181	3505	4304	4710	4711	6356	6359	6362	6363	6491	7096	7372
	7486	7420												
PUSHM	548													
RAL	4556													
RC	3112	3339	4577	6850	7228									
RET	856	886	1038	2267	2457	2525	2653	2980	2993	3083	3100	3107	3119	
	3159	3540	3579	4034	4267	4873	5260	5487	5119	5244	5593	5752	6204	6412
	6506	6813	6886	6865	7058	7076	7230	7256	7290	7352	7393	7556	7794	
RLC	2903	4908	5126											
RNC	797	1856	2169	2965	3178	4582	4813	4776	5380	6858	7222			
RNZ	824	1744	2177	2645	3001	3010	3014	3061	3088	3215	3352	3608	6426	7034
	7304													
RST	1762	1795	1853	1964	1986	2115	2160	2161	2163	2271	2274	2355	2470	2475
	2534	2779	2828	2914	2920	2924	3144	3177	3184	3214	3236	3321	3458	3585
	3514	3518	3594	3659	3700	3723	3746	3836	3865	3868	3881	3944	3976	3982
	4053	4111	4151	4180	4211	4227	4238	4309	4327	4459	4562	4710	4711	4998
	5052	5058	5125	5134	5159	5498	5500	5545	5560	5598	5637	5732	5903	5928
	6094	6150	6203	6319	6327	6329	6356	6359	6362	6363	6389	6394	6435	6491
	6561	6600	6641	6676	6726	6742	6777	6815	6847	6853	723.	7048	7090	7205
	7210	7270	7274	7294	7324	7328	7350	7372	7400	7420	7425	7442	7546	7787
RZ	1768	1798	2158	2197	2929	3402	3404	3722	3751	3876	5372	5376	5499	5581
	5586	5590	6639	6841	6873	6989	7470							
SBB	5224	5402	6267											
SBLC	870	1871	1950	2029	2191	2207	2214	2223	2226	2246	2253	2868	2900	2989
	3013	3030	3036	3079	5241	3368	3473	4523	4375	4436	4521	4559	4572	4886
	4890	4944	4982	4986	5102	5446	5719	5726	5905	5909	6038	6088	6584	6608
	6680	6706	6860	7035	7054	7553								
SPHL	2241	2755	3559	4389	4445									
STA	1880	1944	2263	2291	2408	2608	2650	2734	3042	3097	3104	3861	3873	4043
	4097	4824	4936	5530	5595	5623	6029	6439	6590	7242	7318	7321		

STAX	1797	2394	2428	2456	2454	2456	3529	6991	7469	7503	7543	7793		
SUB	860	831	3221	4423	6264	7179	7225							
SUI	2598	2411	2931	3195	3418	3923	4539	4575	5044	6175	7480			
SYNCHK	42#	2113	2774	3058	3594	3659	3944	4053	4111	5052	5058	5153	5159	5500
	5903	6319	6327	6329	6435	7205	7210	7274	7294	7324	7787			
XCHG	1758	1762	1847	1854	2032	2084	2088	2124	2381	2392	2438	2444	2757	2899
	2923	2947	2956	2984	2996	3075	3082	3157	3201	3240	3535	4202	4252	4322
	4326	4435	5098	5164	5171	5175	5633	5746	5884	5894	5923	5927	6071	6080
	6116	6198	6216	6239	6561	6666	6684	6721	6725	6737	6741	6772	6776	6810
	6814	6956	7020	7052	7190	7293	7337	7422	7431	7537				
XRA	1887	1943	2078	2290	3041	3103	3527	3880	4096	4553	5377	5529	5540	5622
	6270	6759	6825	7870	7126	7282								
XTHL	765	7.0	859	1793	1821	1825	2020	2126	2743	2768	2775	2837	3290	3374
	3471	4108	4117	4206	4236	4403	5168	5177	5187	5879	5891	6147	6333	6354
	6559	6652	6854	6921	6925	6948	6966	6974	7127	7213	7378	7421	7428	7536
	7785													

```
MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 1
C       23-AUG-64 05:46          COMMON FILE

                                00100   SEARCH  MC8808                          ;THE UNIVERSAL FILE
                                00200   SUBTTL  COMMON FILE
                                00300   SALL
                    000001      00400   LENGTH==1                               ; 0 MEANS 4K, 1 MEANS 8K, 2 MEANS 12K
                    000001      00500   REALIO==1
                    000001      00600   CASSW==1                                ;CASSETTE SWITCH (CSAVE,CLOAD)
                    000000      00700   LPTSW==0
                    000000      00800   DSKFJN==0                               ;ON TO READ/WRITE
                    000000      00900   CONSSW==0
                10
                    000001      01100   CONTRW==1                               ;ALLOW ^O
                12                  01200   IFE     REALIO,<
                13                  01300           LPTSW==0                    ;SIMULATOR DEFAULTS
                14                  01400           CASSW==0
                15                  01500           CONSSW==0
                16                  01600           DSKFJN==0
                17                  01700           CONTRW==0>
                18
                19                  01800   IFE     LENGTH,<
                20                  01900           EXTFNC==0                   ; ON MEANS EXTENDED FUNCTIONS
                21                  02000           MULDIM==0                   ; ON MEANS MULTIPLE DIMENSIONED ARRAYS ALLOWED
                22                  02100           STRING==0                   ; ON MEANS STRINGS ALLOWED
                23                  02200           CASSW==0
                24                  02300           LPTSW==0
                25                  02400           DSKFJN==0
                26                  02500           CONSSW==0
                27                  02600           CONTRW==0>
                28
                29                  02700   IFE     LENGTH-1,<
                30      000000      02800           EXTFNC==0
                31      000000      02900           MULDIM==0
                32      000001      03000           STRING==1>
                33
                34                  03100   IFE     LENGTH-2,<
                35                  03200           EXTFNC==1
                36                  03300           MULDIM==1
                37                  03400           STRING==1>
                38
                39                  03500   DEFINE  SYNCHK(A),<MST  1
                40                  03600           A>
                41                  03700   DEFINE  CHRGET,<RST     2>
                42                  03800   DEFINE  OUTCHR,<RST     3>
                43                  03900   DEFINE  COMPAR,<RST     4>
                44                  04000   DEFINE  FSIGN,<RST      5>
                45                  04100   DEFINE  PUSHM,<RST      6>
                46                  04200   DEFINE  PUSHFM,<PUSHM
                47                  04300                   PUSHM>
                48                  04400   DEFINE  ACRLF,<
                49                  04500           "D13
                50                  04600           IFN     STRING,<"D10>>
                51                  04700   DEFINE  PUSHM,<
                52                  04800           PUSH    D
                53                  04900           PUSH    B>
```

```
MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 1-1
C       23-AUG-64 05:46          COMMON FILE

                54                  05000   DEFINE  POPM,<
                55                  05100           POP     B
                56                  05200           POP     D>
                57                  05300   DEFINE  MOVMI(d,C,D,E),<
                58                  05400           XWD     "01000,"0001    ;; "LXI B"
                59                  05500           EXP     C
                60                  05600           EXP     B
                61                  05700           XWD     "01200,"0021    ;; "LXI D"
                62                  05800           EXP     E
                63                  05900           EXP     D>
                64
                65                  06000   IF1,<
                66                  06100   IFE     LENGTH,<PRINTX  /SMALL/>
                67                  06200   IFE     LENGTH-1,<PRINTX        /MEDIUM/>
                68                  06300   IFE     LENGTH-2,<PRINTX        /BIG/>
                69                  06400   IFE     REALIO,<PRINTX  /SIMULATE/>
                70                  06500   IFN     REALIO,<PRINTX  /ON MACHINE/>
                71                  06600   IFN     CASSW,<PRINTX   /CASSETTE/>
                72                  06700   IFN     LPTSW,<PRINTX   /LPT/>
                73                  06800   IFN     DSKFJN,<PRINTX  /DISK/>
                74                  06900   IFN     CONSSW,<PRINTX  /CONSOLE/>>
                75                  07000   PAGE
```

```
                                00100   SUBTTL  FLOATING POINT MATH PACKAGE CONFIGURATION
                                00120   TITLE   MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF
                                00140
                                00160   IFNDEF  .LENGTH,<
                                00180           PRINTX  !!! MUST HAVE COM !!!
                                00200           END>
                                00220
                                00240   RADIX   8                       ;!!!! ALERT !!.!
                                00260                                   ;THROUGHOUT THE MATHPACKAGE..
                                00280
                    000000      00300           .P==0
                                00320
                                00340   INTERNAL        ZERO,FLOAT,FLOATR,MOVE,FADD,FADD8,FSUB,FMULT,FDIV,FIN,FOUT
                                00360   INTERNAL        PJSNF,ABS,INT,GINT,SGN,SQR,RND,SIN,FCOMP,SIGNC,OVERR
                                00380   INTERNAL        INPRT,LINPRT,MOVFM,MOVMF,MOVFR,MOVRF,MOVRM,NEG,INRANT,INXHRT
                                00400   IFN     .EXTFNC,<
                                00420   INTERNAL        FPWR,EXP,LOG,COS,TAN,ATN,FONE>
                                00440   IFN     MULDIM8<.LENGTH-2>,<
                                00460   INTERNAL        QMULT>
                                00480   IFN     STRING,<
                                00500   INTERNAL        SIGNS>
                                00520   IFN     .LENGTH-2,<
                                00540   INTERNAL        FADDT,FSUBT,FMULTT,FDIVT>
                                00560   IFE     .LENGTH-1,<
                                00580   INTERNAL        FPWRT>
                                00600   IFE     .LENGTH-2,<
                                00620   INTERNAL        VMOVMF,VMOVFM,FNCINT,FRCSNG,FRCDBL,VNEG,PJFOUT,OCXBNT,IADD
                                00640   INTERNAL        ISUB,IMULT,IDIV,ICOMP,INEG,DADD,DSUB,DMULT,DDIV,DCOMP,INTFNC>
                                00660
                                00680
                                00700   EXTERNAL        FAC,FACLO,F8_FFR,MINUTK,PLUSTK,ERROR,DV0ERR,ENROV,FCERR,SIGN
                                00720   EXTERNAL        SCODE
                                00740   IFE     .LENGTH-2,<
                                00760   EXTERNAL        DFACLO,ARG,ARGLO,VALTYP,TMERR,TEMP2,TEMP3>
                                01000
                                01020
                                01040   COMMENT %
                                01060           EXTERNAL LOCATIONS USED BY THE MATH-PACKAGE
                                01080           (THE FLOATING ACCUMULATOR
                                01100   IFE     .LENGTH-2,<
                                01120           BLOCK   1               ;(TEMPORARY LEAST SIGNIFICANT BYTE)
                    001920      01140   DFACLO: BLOCK   4>              ;(FOUR LOWEST ORDERS FOR DOUBLE PRECISION)
                    001940      01160   FACLO:  BLOCK   3               ;(LOW ORDER OF MANTISSA (LO))
                    001960      01180                                   ;(MIDDLE ORDER OF MANTISSA (MO))
                    001980      01200                                   ;(HIGH ORDER OF MANTISSA (MO))
                    002000      01220   FAC:    BLOCK   2               ;(EXPONENT)
                    002020      01240                                   ;(TEMPORARY COMPLEMENT OF SIGN IN MSB)
                                01260   IFE     .LENGTH-2,<
                    002040      01280   ARGLO:  BLOCK   7               ;(LOCATION OF SECOND ARGUMENT FOR DOUBLE
                    002080      01300   ARG:    BLOCK   1>              ; PRECISION)
                    002100      01320   FBUFFR: BLOCK   ^D13            ;BUFFER FOR FOUT
                                01340   IFE     .LENGTH-2,<BLOCK ^D<30-13>>
                                01360                                   S
```



```
                                01380
                                01400   THE FLOATING POINT FORMAT IS AS FOLLOWS:
                                01420
                                01440   THE SIGN IS THE FIRST BIT OF THE MANTISSA
                                01460   THE MANTISSA IS 24 BITS LONG
                                01480   THE BINARY POINT IS TO THE LEFT OF THE MSB
                                01500   NUMBER = MANTISSA * 2 ^ EXPONENT
                                01520   THE MANTISSA IS PUSITIVE, WITH A ONE ASSUMED TO BE WHERE THE SIGN BIT IS
                                01540   THE SIGN OF THE EXPONENT IS THE FIRST BIT OF THE EXPONENT
                                01560   THE EXPONENT IS STORED IN EXCESS 200 I.E. WITH A BIAS OF 200
                                01580   SO, THE EXPONENT IS A SIGNED 8-BIT NUMBER WITH 200 ADDED TO IT
                                01600   AN EXPONENT OF ZERO MEANS THE NUMBER IS ZERO, THE OTHER BYTES ARE IGNORED
                                01620   TO KEEP THE SAME NUMBER IN THE FAC WHILE SHIFTING:
                                01640           TO SHIFT RIGHT, EXP:=EXP+1
                                01660           TO SHIFT LEFT,  EXP:=EXP-1
                                01680
                                01700   SO, IN MEMORY THE NUMBER LOOKS LIKE THIS:
                                01720           (BITS 17-24 OF THE MANTISSA)
                                01740           (BITS 9-16 OF THE MANTISSA)
                                01760           (THE SIGN IN BIT 7, BITS 2-8 OF THE MANTISSA ARE IN BITS 6-0)
                                02000           (THE EXPONENT AS A SIGNED NUMBER + 200)
                                02020   (REMEMBER THAT BIT 1 OF THE MANTISSA IS ALWAYS A ONE)
                                02040
                                02060   ARITHMETIC ROUTINE CALLING CONVENTIONS:
                                02080
                                02100   FOR ONE ARGUMENT FUNCTIONS:
                                02120           THE ARGUMENT IS IN THE FAC, THE RESULT IS LEFT IN THE FAC
                                02140   FOR TWO ARGUMENT OPERATIONS:
                                02160           THE FIRST ARGUMENT IS IN B,C,D,E I.E, THE "REGISTERS"
                                02180           THE SECOND ARGUMENT IS IN THE FAC
                                02200           THE RESULT IS LEFT IN THE FAC
                                02220
                                02240   THE "S" ENTRY POINTS TO THE TWO ARGUMENT OPERATIONS HAVE (HL) POINTING TO
                                02260   THE FIRST ARGUMENT INSTEAD OF THE FIRST ARGUMENT BEING IN THE REGISTERS,
                                02280   MOVRM IS CALLED TO GET THE ARGUMENT IN THE REGISTERS.
                                02300   THE "T" ENTRY POINTS ASSUME THE FIRST ARGUMENT IS ON THE STACK,
                                02320   POPR IS USED TO GET THE ARGUMENT IN THE REGISTERS,
                                02340   NOTE: THE "T" ENTRY POINTS SHOULD ALWAYS BE JUMPED TO AND NEVER CALLED
                                02360   BECAUSE THE RETURN ADDRESS ON THE STACK WILL BE CONFUSED WITH THE NUMBER,
                                02380
                                02400   ON THE STACK, THE TWO LO'S ARE PUSHED ON FIRST AND THEN THE MO AND SIGN,
                                02420   THIS IS DONE SO IF A NUMBER IS STORED IN MEMORY, IT CAN BE PUSHED ON THE
                                02440   STACK WITH TWO PUSHM'S.  THE LOWER BYTE OF EACH PAIR IS IN THE LOWER
                                02460   MEMORY ADDRESS SO WHEN THE NUMBER IS POPPED INTO THE REGISTERS, THE HIGHER
                                02480   ORDER BYTE WILL BE IN THE HIGHER ORDER REGISTER OF THE REGISTER PAIR, I.E,
                                02500   THE HIGHER ORDER BYTE WILL BE POPPED INTO B, D OR H.
                                02520   %
                                02540   PAGE
```

```
#0      MAC     23-AUG-84 06108         FLOATING POINT ADDITION AND SUBTRACTION

                                        02120   SUBTTL FLOATING POINT ADDITION AND SUBTRACTION
                                        02140           ;ENTRY TO FADD WITH POINTER TO ARG IN (HL)
000000' 021000  000041          02160   FADDM:  LXI     H,FHALF         ;ENTRY TO ADD 1/2
000001' 000000  002312'
000002' 000000  000000
000003' 021000  000315          02180   FADD3:  CALL    MOVFM           ;GET ARGUMENT INTO THE REGISTERS
000004' 000000  001245'
000005' 020000  000001'
000006' 001000  000303          02200           JMP     FADU            ;DO THE ADDITION
000007' 000000  000025'
000010' 000000  000004'

                                        02260           ;SUBTRACTION    FAC:=ARG-FAC
                                        02280   IFN     EXTFNC,<
000011' 021000  000315          02300   FSUBM:  CALL    MOVFRM>         ;ENTRY IF POINTER TO ARG IS IN (HL)
000012' 000000  001243'
000013' 000000  000007'
                                        02320   IFE     LENGTH-1,<
000014' 021000  000041          02340           XWD     1000,041>       ;"LXI   H" AROUND NEXT 2 BYTES
                                        02360   IFN     LENGTH-2,<
000015' 021000  000381          02380   FSUBI:  PUSH>                   ;ENTRY IF ARGUMENT IS ON THE STACK
000016' 021000  000321
000017' 021300  000315          02400   FSUB:   CALL    NEG             ;NEGATE SECOND ARGUMENT
000020' 000000  001175'
000021' 000000  000012'
                                        02420                           ;FALL INTO FADD

                                        02460           ;ADDITION       FAC:=ARG+FAC
                                        02500           ;ALTERS A,B,C,D,E,H,L
                                        02520   IFN     LENGTH-2,<
000022' 001000  000041          02540           XWD     1000,041        ;"LXI   H" AROUND NEXT 2 BYTES
000023' 001000  000301          02560   FADDI:  PUSH>                   ;ENTRY IF ARGUMENT IS ON THE STACK
000024' 001000  000321
000025' 021000  000170          02580   FADU:   MOV     A,B             ;CHECK IF FIRST ARGUMENT IS ZERO
000026' 001000  000267          02600           ORA     A               ;GET EXPONENT
000027' 021000  000310          02620           RZ                      ;IT IS, RESULT IS NUMBER IN FAC
000030' 001000  000072          02640           LDA     FAC             ;GET EXPONENT
000031' 000000  000000*
000032' 000000  000020*
000033' 001000  000267          02660           ORA     A               ;SEE IF THE NUMBER IS ZERO
000034' 001000  000312          02680           JZ      MOVFR           ;IT IS, ANSWER IS IN REGISTERS
000035' 000000  001225'
000036' 000000  000303'

                                        02720   ;WE WANT TO GET THE SMALLER NUMBER IN THE REGISTERS SO WE CAN SHIFT IT RIGHT
                                        02740   ;AND ALIGN THE BINARY POINTS OF THE TWO NUMBERS.  THEN WE CAN JUST ADD OR
                                        02760   ;SUBTRACT THEM (DEPENDING ON THEIR SIGNS) BYTEWISE.
000037' 001000  000220          02780           SUB     B               ;CHECK RELATIVE SIZES
000040' 001000  000322          02800           JNC     FADU1           ;IS FAC SMALLER?
000041' 000000  000000'
000042' 000000  000035'
```



```
#4      MAC     23-AUG-84 06108         FLOATING POINT ADDITION AND SUBTRACTION

000043' 021000  000057          02820           CMA                     ;YES, NEGATE SHIFT COUNT
000044' 021000  000074          02840           INR     A
000045' 021200  000353          02860           XCHG                    ;SWITCH FAC AND REGISTERS, SAVE (DE)
000046' 021000  000315          02880           CALL    PUSHF           ;PUT FAC ON STACK
000047' 000000  001203'
000050' 000000  000041'
000051' 021000  000353          02900           XCHG                    ;GET (DE) BACK WHERE IT BELONGS
000052' 021000  000315          02920           CALL    MOVFR           ;PUT REGISTERS IN THE FAC
000053' 000000  001225'
000054' 000000  000047'
000055' 021000  000301          02940           POP     H               ;GET THE OLD FAC IN THE REGISTERS
000056' 021000  000321
000057'                         02960   FADD1:
                                        02980   IFN     LENGTH,<
000057' 001000  000376          03000           CPI     31              ;ARE WE WITHIN 24 BITS?
000060' 000000  000031
000061' 001000  000320          03020           RNC>                    ;NO, ALL DONE
000062' 001000  000365          03040           PUSH    PSW             ;SAVE SHIFT COUNT
000063' 001000  000315          03060           CALL    UNPACK          ;UNPACK THE NUMBERS
000064' 000000  001272'
000065' 000000  000053'
000066' 001000  000147          03080           MOV     M,A             ;SAVE SUBTRACTION FLAG
000067' 001000  000361          03100           POP     PSW             ;GET SHIFT COUNT BACK
000070' 001000  000315          03120           CALL    SHIFTR          ;SHIFT REGISTERS RIGHT THE RIGHT AMOUNT
000071' 000000  000334'
000072' 000000  000064'

                                        03160   ;IF THE NUMBERS HAVE THE SAME SIGN, THEN WE ADD THEM.  IF THE SIGNS ARE
                                        03180   ;DIFFERENT, THEN WE HAVE TO SUBTRACT THEM.  WE HAVE TO DO THIS BECAUSE THE
                                        03200   ;MANTISSAS ARE POSITIVE.  JUDGING BY THE EXPONENTS, THE LARGER NUMBER IS IN
                                        03220   ;THE FAC, SO IF WE SUBTRACT, THE SIGN OF THE RESULT SHOULD BE THE SIGN OF THE
                                        03240   ;FAC!  HOWEVER, IF THE EXPONENTS ARE THE SAME, THE NUMBER IN THE REGISTERS
                                        03260   ;COULD BE BIGGER, SO AFTER WE SUBTRACT THEM, WE HAVE TO CHECK IF THE RESULT
                                        03280   ;WAS NEGATIVE.  IF IT WAS, WE NEGATE THE NUMBER IN THE REGISTERS AND
                                        03300   ;COMPLEMENT THE SIGN OF THE FAC.  (HERE THE FAC IS UNPACKED)
                                        03320   ;IF WE HAVE TO ADD THE NUMBERS, THE SIGN OF THE RESULT IS THE SIGN OF THE
                                        03340   ;FAC,  SO, IN EITHER CASE, WHEN WE ARE ALL DONE, THE SIGN OF THE RESULT
                                        03360   ;WILL BE THE SIGN OF THE FAC.
000073' 001000  000264          03380           ORA     M               ;GET SUBTRACTION FLAG
000074' 001000  000041          03400           LXI     H,FACLO         ;SET POINTER TO LO'S
000075' 000000  000006*
000076' 000000  000071*
000077' 001000  000362          03420           JP      FADD3           ;SUBTRACT IF THE SIGNS WERE DIFFERENT
000100' 000000  000025'
000101' 000000  000075'
000102' 001000  000315          03440           CALL    FADDA           ;ADD THE NUMBERS
000103' 000000  000074'
000104' 000000  000100'
000105' 001000  000322          03460           JNC     ROUND           ;ROUND RESULT IF THERE WAS NO OVERFLOW
000106' 000000  000033'
000107' 000000  000103'
                                        03480                           ;THE MOST IT CAN OVERFLOW IS ONE BIT
000110' 001000  000043          03500           INX     H               ;THERE WAS OVERFLOW
```

```
F4      MAC     23-AUG-64 06:08         FLOATING POINT ADDITION AND SUBTRACTION

                                        03520           INR     M               ;INCREMENT EXPONENT
                                        03540           JZ      OVERR           ;CHECK FOR OVERFLOW
                                        03560
                                        03580   IFE     LENGTH,<
                                        03600           CALL    SHFTRO>         ;SHIFT RESULT RIGHT ONE, SHIFT CARRY IN
                                        03620   IFN     LENGTH,<
                                        03640           MVI     L,1             ;SHIFT RESULT RIGHT ONE, SHIFT CARRY IN
                                        03660           CALL    SHRADO>
                                        03680           JMP     ROUND           ;ROUND RESULT AND WE ARE DONE
                                        03700
                                        03720           ;HERE TO SUBTRACT C,D,E,B FROM ((HL)+0,1,2),0
000125' 001000  000257  03740   FADD3:  XRA     A               ;SUBTRACT NUMBERS, NEGATE UNDERFLOW BYTE
000126' 001000  000220  03760           SUB     B
000127' 001000  000127  03780           MOV     B,A             ;SAVE IT
000130' 001000  000176  03800           MOV     A,M             ;SUBTRACT LOW ORDERS
000131' 001000  000233  03820           SBB     E
000132' 001000  000137  03840           MOV     E,A
000133' 001000  000043  03860           INX     H               ;UPDATE POINTER TO NEXT BYTE
000134' 001000  000176  03880           MOV     A,M             ;SUBTRACT MIDDLE ORDERS
000135' 001000  000232  03900           SBB     D
000136' 001000  000127  03920           MOV     D,A
000137' 001000  000043  03940           INX     H               ;UPDATE POINTER TO HIGH ORDERS
000140' 001000  000176  03960           MOV     A,M             ;SUBTRACT HIGH ORDERS
000141' 001000  000231  03980           SBB     C
000142' 001000  000117  04000           MOV     C,A
                                        04020           ;BECAUSE WE WANT A POSITIVE MANTISSA, CHECK IF WE HAVE TO NEGATE THE
                                        04040           ; NUMBER
000143' 001200  000334  04060   FADFLT: CC      NEGR            ;ENTRY FROM FLOATR, INT: NEGATE NUMBER IF IT
000144' 000000  000310'
000145' 000000  000123'
                                        04080                   ; WAS NEGATIVE, FALL INTO NORMALIZE
                                        04100
                                        04120           ;NORMALIZE C,D,E,B
                                        04140           ;ALTERS A,B,C,D,E,H,L
                                        04160           ;HERE WE SHIFT THE MANTISSA LEFT UNTIL THE MSB IS A ONE,
                                        04180           ;EXCEPT IN 4K, THE IDEA IS TO SHIFT LEFT BY 8 AS MANY TIMES AS
                                        04200           ;POSSIBLE.
000146'                 04220   NORMAL:
                                        04240   IFE     LENGTH,<
                                        04260           MVI     H,0             ;CLEAR SHIFT COUNT
                                        04280           MOV     A,C             ;IS THE NUMBER NORMALIZED?
                                        04300           ORA     A
                                        04320           JM      ROUND           ;YES, WE ARE DONE
                                        04340   NORM2:  CPI     340             ;IS THE RESULT ZERO?
                                        04360           JZ      ZERO            ;YES, ZERO THE FAC
                                        04380           DCR     H               ;NO, DECREMENT SHIFT COUNT
                                        04400           MOV     A,B             ;SHIFT THE LO LEFT
```



```
F4      MAC     23-AUG-64 06:08         FLOATING POINT ADDITION AND SUBTRACTION

                                        04420           ADD     A               ;SHIFT IN A ZERO
                                        04440           MOV     B,A
                                        04460           CALL    SHFTLO          ;SHIFT THE REST OF THE NUMBER LEFT ONE
                                        04480           MOV     A,H             ;GET THE SHIFT COUNT
                                        04500           JP      NORM2>          ;CONTINUE IF NUMBER IS NOT NORMALIZED
                                        04520   IFN     LENGTH,<
000146' 001000  000150  04540           MOV     L,B             ;PUT LOWEST 2 BYTES IN (HL)
000147' 001000  000143  04560           MOV     H,E
000150' 001000  000257  04580           XRA     A               ;ZERO SHIFT COUNT
000151' 001000  000107  04600   NORM1:  MOV     B,A             ;SAVE SHIFT COUNT
000152' 001000  000171  04620           MOV     A,C             ;DO WE HAVE 1 BYTE OF ZEROS
000153' 001000  000267  04640           ORA     A
000154' 001000  000302  04660           JNZ     NORM3           ;NO, SHIFT ONE PLACE AT A TIME
000155' 000000  000210'
000156' 000000  000144'
                                        04680           ;THIS LOOP SPEEDS THINGS UP BY SHIFTING 8 PLACES AT ONE TIME
000157' 001000  000112  04700           MOV     C,D             ;YES, SHIFT OVER 1 BYTE
000160' 001000  000124  04720           MOV     D,H
000161' 001000  000145  04740           MOV     H,L
000162' 001000  000157  04760           MOV     L,A             ;SHIFT IN 8 ZEROS FOR THE LOW ORDER
000163' 001000  000170  04780           MOV     A,B             ;UPDATE SHIFT COUNT
000164' 001000  000326  04800           SUI     10
000165' 000000  000010
000166' 001000  000376  04820           CPI     340             ;DID WE SHIFT IN 4 BYTES OF ZEROS?
000167' 000000  000340
000170' 001000  000302  04840           JNZ     NORM1           ;NO, TRY TO SHIFT OVER 8 MORE
000171' 000000  000151'
000172' 000020  000155'
                                        04860                   ;YES, NUMBER WAS ZERO.  FALL INTO ZERO
                                        04880
                                        04900           ;ZERO FAC
                                        04920           ;ALTERS A ONLY
                                        04940           ;EXITS WITH A=0
                                        04960           ;BY OUR FLOATING POINT FORMAT, THE NUMBER IS ZERO IF THE EXPONENT IS
                                        04980           ; ZERO
000173' 001000  000257  05000   ZERO:   XRA     A               ;ZERO A
000174' 001000  000066  05020   ZERO0:  STA     FAC             ;ZERO THE FAC'S EXPONENT, ENTRY IF A=0
000175' 000000  000031'
000176' 000000  000171'
000177' 001000  000311  05040           RET                     ;ALL DONE
                                        05060
                                        05080
000200' 001000  000005  05100   NORM2:  DCR     B               ;DECREMENT SHIFT COUNT
000201' 001000  000151  05120           DAD     H               ;ROTATE (HL) LEFT ONE, SHIFT IN A ZERO
000202' 001000  000172  05140           MOV     A,D             ;ROTATE NEXT HIGHER ORDER LEFT ONE
000203' 001000  000027  05160           RAL
000204' 001000  000127  05180           MOV     D,A
000205' 001000  000171  05200           MOV     A,C             ;ROTATE HIGH ORDER LEFT ONE
000206' 001000  000217  05220           ACC     A               ;SET CONDITION CODES
000207' 001000  000117  05240           MOV     C,A
000210' 001000  000362  05260   NORM3:  JP      NORM2           ;WE HAVE MORE NORMALIZATION TO DO
000211' 000000  000200'
```

```
                                                                ;ALL NORMALIZED, GET SHIFT COUNT
                                                                ;PUT LO'S BACK IN E,B
                                                                ;CHECK IF WE DID NO SHIFTING
                                                                ;LOOK AT FAC'S EXPONENT
                                                                ;UPDATE EXPONENT
                                                                ;CHECK FOR UNDERFLOW
                                                                ;NUMBER IS ZERO, ALL DONE
                                                                ;FALL INTO ROUND AND WE ARE DONE

                                                                ;ROUND RESULT IN C,D,E,B AND PUT NUMBER IN THE FAC
                                                                ;ALTERS A,B,C,D,E,H,L
                                                                ;WE ROUND C,D,E UP OR DOWN DEPENDING UPON THE MSB OF B
                                                                ;SEE IF WE SHOULD ROUND UP
                                                                ;ENTRY FROM FDIV, GET POINTER TO EXPONENT
                                                                ;DO IT IF NECESSARY

                                                                ;PUT EXPONENT IN B
                                                                ;HERE WE PACK THE MQ AND SIGN
                                                                ;POINT TO SIGN
                                                                ;GET SIGN
                                                                ;GET RID OF UNWANTED BITS
                                                                ;PACK SIGN AND MQ
                                                                ;SAVE IT IN C
                                                                ;SAVE NUMBER IN FAC

                                                                ;SHIFT C,D,E LEFT ONE
                                                                ;THIS IS USED BY NORMAL, FDIV
                                                                ;ALTERS A,C,D,E
                                                                ;GET THE LO
                                                                ;SHIFT IT
                                                                ;SAVE IT
                                                                ;SHIFT THE NEXT HIGHER ORDER
```



```
                                                                ;SHIFT THE HIGHEST ORDER
                                                                ;ROTATE A LEFT AND SET CONDITION CODES
                                                                ;ALL DONE

                                                                ;SUBROUTINE FOR ROUND: ADD ONE TO C,D,E
                                                                ;ADD ONE TO THE LOW ORDER, ENTRY FROM QINT
                                                                ;ALL DONE IF IT IS NOT ZERO
                                                                ;ADD ONE TO NEXT HIGHER ORDER
                                                                ;ALL DONE IF NO OVERFLOW
                                                                ;ADD ONE TO THE HIGHEST ORDER
                                                                ;RETURN IF NO OVERFLOW
                                                                ;THE NUMBER OVERFLOWED, SET NEW HIGH ORDER

                                                                ;UPDATE EXPONENT
                                                                ;RETURN IF IT DID NOT OVERFLOW
                                                                ;IT DID, FALL INTO OVERR

                                                                ;OVERFLOW ERROR
OVERR:  MVI     E,ERRDV         ;SET OVERFLOW ERROR CODE
        JMP     ERROR           ;GO TO ITII

                                                                ;ADD (HL)+2,1,0 TO C,D,E
                                                                ;THIS CODE IS USED BY FADD, FOUT
FADDA:  MOV     A,M             ;GET LOWEST ORDER
        ADD     E               ;ADD IN OTHER LOWEST ORDER
        MOV     E,A             ;SAVE IT
        INX     H               ;UPDATE POINTER TO NEXT BYTE
        MOV     A,M             ;ADD MIDDLE ORDERS
        ADC     D
        MOV     D,A
        INX     H               ;UPDATE POINTER TO HIGH ORDER
        MOV     A,M             ;ADD HIGH ORDERS
        ADC     C
        MOV     C,A
        RET                     ;ALL DONE

                                                                ;NEGATE NUMBER IN C,D,E,B
                                                                ;THIS CODE IS USED BY FADD, QINT
                                                                ;ALTERS A,B,C,D,E,L
NEGN:   LXI     H,FAC+1         ;NEGATE FAC

        MOV     A,M             ;GET SIGN
        CMA                     ;COMPLEMENT IT
        MOV     M,A             ;SAVE IT AGAIN
```

```
F#      MAC     23-AUG-64 06108         FLOATING POINT ADDITION AND SUBTRACTION

   495  000316' 001000  000257          07000           XRA     A               ;ZERO A
   496  000317' 001000  000157          07020           MOV     L,A             ;SAVE ZERO IN L
   497  000320' 001000  000220          07040           SUB     B               ;NEGATE LOWEST ORDER
   498  000321' 001000  000107          07060           MOV     B,A             ;SAVE IT
   499  000322' 001000  000175          07080           MOV     A,L             ;GET A ZERO
   500  000323' 001000  000233          07100           SBB     E               ;NEGATE NEXT HIGHEST ORDER
   501  000324' 001000  000137          07120           MOV     E,A             ;SAVE IT
   502  000325' 001000  000175          07140           MOV     A,L             ;GET A ZERO
   503  000326' 001000  000232          07160           SBB     D               ;NEGATE NEXT HIGHEST ORDER
   504  000327' 001000  000127          07180           MOV     D,A             ;SAVE IT
   505  000330' 001000  000175          07200           MOV     A,L             ;GET ZERO BACK
   506  000331' 001000  000231          07220           SBB     C               ;NEGATE HIGHEST ORDER
   507  000332' 001000  000117          07240           MOV     C,A             ;SAVE IT
   508  000333' 001000  000311          07260           RET                     ;ALL DONE
   509
   510
   511                                  07320           ;SHIFT C,D,E RIGHT
   512                                  07340           ;A = SHIFT COUNT
   513                                  07360           ;ALTERS A,B,C,D,E,L
   514                                  07380           ;THE IDEA (EXCEPT IN 4K) IS TO SHIFT RIGHT 8 PLACES AS MANY TIMES AS
   515                                  07400           ; POSSIBLE
   516  000334' 001000  000006          07420   SHIFTR: MVI     B,8             ;ZERO OVERFLOW BYTE
   517  000335' 000000  000000          07440           IFE     LENGTH,<
   518                                  07460           INR     A>              ;ADD ONE TO SHIFT COUNT
   519                                  07480           IFN     LENGTH,<
   520  000336' 001000  000526          07500   SHFTR1: SUI     10              ;CAN WE SHIFT IT 8 RIGHT?
   521  000337' 002000  000010          07520           JC      SHFTR2          ;NO, SHIFT IT ONE PLACE AT A TIME
   522  000340' 001000  000332          07540           ;THIS LOOP SPEEDS THINGS UP BY SHIFTING 8 PLACES AT ONE TIME
   523  000341' 000000  000355'         07560           MOV     B,E             ;SHIFT NUMBER 1 BYTE RIGHT
   524  000342' 002000  000311'         07580           MOV     E,D
   525  000343' 001000  000132          07600           MOV     D,C
   526                                  07620           MVI     C,0             ;PUT 0 IN MO
   527  000345' 001000  000121          07640           JMP     SHFT#1          ;TRY TO SHIFT 8 RIGHT AGAIN
   528  000346' 001000  000016
   529  000347' 000000  000000
   530  000350' 001000  000303
   531  000351' 000000  000336'
   532  000352' 000000  000341'
   533  000353' 001000  000306          07660   SHFTR2: ADI     11>             ;CORRECT SHIFT COUNT
   534  000354' 000000  000011
   535  000355' 001000  000157          07680           MOV     L,A             ;SAVE SHIFT COUNT
   536  000356' 001000  000257          07700   SHFTR3: XRA     A               ;CLEAR CARRY
   537  000357' 001000  000355          07720           DCR     L               ;ARE WE DONE SHIFTING?
   538  000360' 001000  000310          07740           RZ                      ;RETURN IF WE ARE
   539                                  07760   IFE     LENGTH,<
   540                                  07780   $HRADD: CALL    $MFTR0>         ;SHIFT THE NUMBER RIGHT ONE
   541                                  07800   IFN     LENGTH,<
   542  000361' 001000  000171          07820           MOV     A,C             ;GET MO
   543  000362' 001000  000037          07840   $HRADD: RAR                     ;ENTRY FROM FADD, SHIFT IT RIGHT
   544  000363' 001000  000117          07860           MOV     C,A             ;SAVE IT
   545  000364' 001000  000172          07880           MOV     A,D             ;SHIFT NEXT BYTE RIGHT
```



```
F#      MAC     23-AUG-64 06108         FLOATING POINT ADDITION AND SUBTRACTION

   546  000365' 001000  000037          07900           RAR
   547  000366' 001000  000127          07920           MOV     D,A
   548  000367' 001000  000173          07940           MOV     A,E             ;SHIFT LOW ORDER RIGHT
   549  000370' 001000  000037          07960           RAR
   550  000371' 001000  000137          07980           MOV     E,A
   551  000372' 001000  000170          08000           MOV     A,B             ;SHIFT OVERFLOW BYTE RIGHT
   552  000373' 001000  000037          08020           RAR
   553  000374' 001000  000107          08040           MOV     B,A>
   554  000375' 001000  000303          08060           JMP     SHFTR3          ;SEE IF WE ARE DONE
   555  000376' 000000  000356'
   556  000377' 000000  000351'
   557
   558
   559                                  08120   IFE     LENGTH,<
   560                                  08140           ;SHIFT C,D,E,B RIGHT ONE
   561                                  08160           ;THIS IS USED BY SHIFTR, FMULT, FADD
   562                                  08180           ;ALTERS A,B,C,D,E
   563                                  08200   $MFTR1: MOV     A,C             ;GET THE MO
   564                                  08220   $MFR0A: RAR                     ;SHIFT IT RIGHT, ENTRY FROM FMULT
   565                                  08240           MOV     C,A
   566                                  08260           MOV     A,D             ;SHIFT THE MO RIGHT
   567                                  08280           RAR
   568                                  08300           MOV     D,A
   569                                  08320           MOV     A,E             ;SHIFT THE LO
   570                                  08340           RAR
   571                                  08360           MOV     E,A
   572                                  08380           MOV     A,B             ;SHIFT THE EXTRA LO BYTE
   573                                  08400           RAR
   574                                  08420           MOV     B,A
   575                                  08440           RET>                    ;ALL DONE
   576                                  08460   PAGE
   577
   578
```

```
                                                    08460   SUBTTL  NATURAL LOG FUNCTION
                                                    08500   IFN     EXTFNC,<
                                                    08520           ;CALCULATION IS BY:
                                                    08540           ; LN(F*2^N)=(N*LOG2(F))*LN(2)
                                                    08560           ;AN APPROXIMATION POLYNOMIAL IS USED TO CALCULATE LOG2(F)
                                                    08580
                                                    08600           ;CONSTANTS USED BY LOG
000400' 000003  000002                              08620   FONE:   000     ; 1
000401' 000003  000003                              08640           000
000402' 000000  000000                              08660           000
000403' 000000  000201                              08680           201
000404' 000000  000003                              08700   LOGCN2: 3       ;DEGREE+1
000405' 000003  000252                              08720           252     ; 0.598978650
000406' 000000  000126                              08740           126
000407' 000000  000031                              08760           031
000410' 000000  000200                              08780           200
000411' 000000  000361                              08800           361     ; 0.961470632
000412' 000000  000042                              08820           042
000413' 000000  000166                              08840           166
000414' 000000  000200                              08860           200
000415' 000000  000105                              08880           105     ; 2.88539129
000416' 000000  000252                              08900           252     ; NOTE: THE REFERENCE FOR THIS CONSTANT HAS 108 NOT 105
000417' 000000  000070                              08920           070     ; IN THE LOW ORDER BYTE.
000420' 000000  000202                              08940           202
                                                    08960
000421' 001000  000357                              08980   LOG:    FSIGN                   ;CHECK FOR A NEGATIVE OR ZERO ARGUMENT
000422' 001000  000352                              09000           JPE     FCERR           ;FAC ,LE. 0, BLOW HIM OUT OF THE WATER
000423' 000002  000002
000424' 000P00  000376'
                                                    09020                                   ;FSIGN ONLY RETURNS 0,1 OR 377 IN A
                                                    09040                                   ;THE PARITY WILL BE EVEN IF A HAS 0 OR 377
000425' 001000  000041                              09060           LXI     H,FAC           ;GET POINTER TO EXPONENT
000426' 000000  000235'
000427' 000000  000423'
000430' 001000  000176                              09080           MOV     A,M             ;GET EXPONENT IN A
000431' 001000  000001                              09100           MOVRI   200,065,204,363 ;GET SQR(.5)
000432' 000000  000065
000433' 000000  000200
000434' 001000  000021
000435' 000000  000363
000436' 000000  000004
000437' 001000  000220                              09120           SUB     B               ;REMOVE EXCESS 200
000440' 001000  000365                              09140           PUSH    PSW             ;SAVE EXPONENT FOR LATER
000441' 001000  000160                              09160           MOV     M,B             ;GET EXP TO 200, RESULT IS NUM IN (.5,1)
000442' 001000  000325                              09180           PUSHR                   ;SAVE SQR(.5)
000443' 001000  000503
000444' 001000  000315                              09200           CALL    FADD            ;CALCULATE (F+SQR(.5))/(F*SQR(.5))
000445' 000000  000025'
000446' 000000  000420'
000447' 001000  000301                              09220           POPR                    ;GET SQR(.5) BACK
000450' 001000  000321
000451' 001000  000004                              09240           INR     B               ;GET SQR(2)
000452' 001000  000315                              09260           CALL    FDIV            ;WHERE F=NUMBER LEFT IN FAC
```



```
000453' 000000  000655'
000454' 000000  000445'
000455' 001000  000041                              09280           LXI     H,FONE          ;THE CALCULATION IS EQUIVALENT TO THE ABOVE,
000456' 000000  000400'
000457' 000000  000453'
000460' 001000  000315                              09300           CALL    FSUBS           ; BUT DONE IN A DIFFERENT ORDER
000461' 000000  000611'
000462' 000000  000456'
000463' 001000  000041                              09320           LXI     H,LOGCN2        ;EVALUATE APPROXIMATION POLYNOMIAL
000464' 000000  000404'
000465' 000000  000461'
000466' 001000  000315                              09340           CALL    POLYX
000467' 000000  000515'
000470' 000000  000464'
000471' 001000  000001                              09360           MOVRI   200,202,000,000 ;GET =1/2
000472' 000000  000200
000473' 000000  000000
000474' 001000  000021
000475' 000000  000000
000476' 000000  000000
000477' 001000  000315                              09380           CALL    FADD            ;ADD IN LAST CONSTANT
000500' 000000  000025'
000501' 000000  000467'
000502' 001000  000361                              09400           POP     PSW             ;RETRIEVE ORIGINAL EXPONENT
000503' 001000  000315                              09420           CALL    FINLOG          ;ADD IT TO ORIGINAL NUMBER
000504' 000000  001731'
000505' 000000  000502'
000506' 001000  000001                              09440   MULLN2: MOVRI   200,261,162,030>;GET LN(2)
000507' 000000  000261
000510' 000000  000200
000511' 001000  000021
000512' 000000  000030
000513' 000000  000162
                                                    09460   ;       JMP     FMULT           ;MULTIPLY BY LN(2)
                                                    09480   PAGE
```

```
                                        09500  SUBTTL  FLOATING MULTIPLICATION AND DIVISION
                                        09520          ;MULTIPLICATION          FAC=ARG*FAC
                                        09540          ;ALTERS A,B,C,D,E,H,L
                                        09560  IFE     EXTFNC,<
                                        09580  FMULTS: CALL    MOVRM>          ;ENTRY WITH POINTER TO ARG IN (HL)
                                        09600  IFN     LENGTH=2,<
                                        09620          XND     1000,041        ;"LXI   H" AROUND NEXT 2 BYTES
                                        09640  FMULT1: POPR>                   ;ENTRY IF ARGUMENT IS ON THE STACK
                                        09660  FMULTS: FSIGN                   ;CHECK IF FAC IS ZERO
                                        09680          RZ                      ;IF IT IS, RESULT IS ZERO
                                        09700          MVI     C,0             ;ADD THE TWO EXPONENTS, L IS A FLAG
                                        09720          CALL    MULDIV          ;FIX UP THE EXPONENTS
                                        09740                  ;SAVE THE NUMBER IN THE REGISTERS SO WE CAN ADD IT FAST
                                        09760          MOV     A,C             ;GET MO
                                        09780          STA     FMULTA+1        ;STORE MO OF REGISTERS
                                        09800          XCHG                    ;STORE THE TWO LO'S OF THE REGISTERS
                                        09820          SHLD    FMULTB+1
                                        09840          LXI     B,$CODE         ;ZERO THE PRODUCT REGISTERS
                                        09860          MOV     D,B
                                        09880          MOV     E,B
                                        09900          LXI     H,NORMAL        ;PUT ADDRESS OF NORMAL, WERE WE FINISH UP,
                                        09920          PUSH    H               ; ON THE STACK
                                        09940          LXI     H,FMULT2        ;PUT FMULT2 ON THE STACK TWICE, SO AFTER
                                        09960          PUSH    H               ; WE MULTIPLY BY THE LO BYTE, WE WILL
                                        09980          PUSH    H               ; MULTIPLY BY THE MO AND MO
                                        10000          LXI     H,FACLO         ;GET ADDRESS OF LO OF FAC
                                        10020  FMULT2: MOV     A,M             ;GET BYTE TO MULTIPLY BY
                                        10040          INX     H               ;MOVE POINTER TO NEXT BYTE
                                        10060  IFN     LENGTH,<
                                        10080          ORA     A
                                        10100          JZ      FMULT3>         ;ARE WE MULTIPLYING BY ZERO?
                                        10120          PUSH    H               ;SAVE POINTER
                                        10140  IFE     LENGTH,<
                                        10160          MVI     L,10>           ;SET UP A COUNT
                                        10180  IFN     LENGTH,<
```



```
                                        10200          XCHG                    ;GET LO'S IN (HL)
                                        10220          MVI     E,10>           ;SET UP A COUNT
                                        10240
                                        10260  ;THE PRODUCT WILL BE FORMED IN C,D,E,B, THIS WILL BE IN C,H,L,B PART OF THE
                                        10280  ;TIME IN ORDER TO USE THE "DAD" INSTRUCTION,  AT FMULT2, WE GET THE NEXT
                                        10300  ;BYTE OF THE MANTISSA IN THE FAC TO MULTIPLY BY,  ((HL) POINTS TO IT)
                                        10320  ;(THE FMULT2 SUBROUTINE PRESERVES (HL))  IN 8K, IF THE BYTE IS ZERO, WE JUST
                                        10340  ;SHIFT THE PRODUCT 8 RIGHT.  THIS BYTE IS THEN SHIFTED RIGHT AND SAVED IN D
                                        10360  ;(H IN 4K),  THE CARRY DETERMINES IF WE SHOULD ADD IN THE SECOND FACTOR
                                        10380  ;IF WE DO, WE ADD IT TO C,H,L.  B IS ONLY USED TO DETERMINE WHICH WAY WE
                                        10400  ;ROUND.  WE THEN SHIFT C,H,L,B  (C,D,E,B) IN 4K RIGHT ONE TO GET READY FOR THE
                                        10420  ;NEXT TIME THROUGH THE LOOP.  NOTE THAT THE CARRY IS SHIFTED INTO THE MSB OF
                                        10440  ;C.  E HAS A COUNT (= IN 4K) TO DETERMINE WHEN WE HAVE LOOKED AT ALL THE BITS
                                        10460  ;OF D (H IN 4K),
                                        10480  FMULT3: RAR                     ;ROTATE BYTE RIGHT
                                        10500  IFE     LENGTH,<
                                        10520          MOV     H,A>            ;SAVE THE COUNT
                                        10540  IFN     LENGTH,<
                                        10560          MOV     D,A>            ;SAVE IT
                                        10580          MOV     A,C             ;GET MO
                                        10600          JNC     FMULT5          ;DON'T ADD IN NUMBER IF BIT WAS ZERO
                                        10620  IFE     LENGTH,<
                                        10640          XCHG>                   ;PUT THE LO'S IN (HL)
                                        10660          PUSH    D               ;SAVE COUNTERS
                                        10680  FMULTB: LXI     D,$CODE         ;GET LO'S OF NUMBER TO ADD, THIS IS SET ABOVE
                                        10700          DAD     D               ;ADD THEM IN
                                        10720          POP     D               ;GET COUNTERS BACK
                                        10740  FMULTA: ACI     0               ;ADD IN MO, THIS IS SET UP ABOVE
                                        10760  IFE     LENGTH,<
                                        10780          XCHG                    ;PUT THE LO'S BACK IN (DE)
                                        10800  FMULT5: CALL    SHFRGA          ;SHIFT THE RESULT RIGHT ONE
                                        10820          DCR     L               ;ARE WE DONE?
                                        10840          MOV     A,M>            ;GET NUMBER WE ARE MULTIPLYING BY
                                        10860  IFN     LENGTH,<
                                        10880  FMULT5: RAR                     ;ROTATE RESULT RIGHT ONE
                                        10900          MOV     C,A
                                        10920          MOV     A,H             ;ROTATE NEXT BYTE
                                        10940          RAR
                                        10960          MOV     H,A
                                        10980          MOV     A,L             ;ROTATE NEXT LOWER ORDER
                                        11000          RAR
                                        11020          MOV     L,A
                                        11040          MOV     A,B             ;ROTATE LO
                                        11060          RAR
                                        11080          MOV     B,A
                                        11100          DCR     E               ;ARE WE DONE?
                                        11120          MOV     A,D>            ;GET NUMBER WE ARE MULTIPLYING BY
```

```
000624' 001000  000302          11140           JNZ     FMULT4          ;MULTIPLY AGAIN IF WE ARE NOT DONE
000625' 000000  000571'
000626' 000000  000601'
                                11160   IFN     LENGTH,<
                                11180           XCHG>                   ;GET LO'S IN (HL)
000627' 001000  000353          11200   POPMRT: POP     H               ;GET POINTER TO NUMBER TO MULTIPLY BY
000630' 001000  000341          11220           RET                     ;ALL DONE
000631' 001000  000311          11240   IFN     LENGTH,<
                                11260   FMULT3: MOV     B,E             ;MULTIPLY BY ZERO: SHIFT EVERYTHING 8 RIGHT
000632' 001000  000103          11280           MOV     E,D
000633' 001000  000132          11300           MOV     D,C
000634' 001000  000121          11320           MOV     C,A             ;SHIFT IN 8 ZEROS ON THE LEFT
000635' 001000  000117          11340           RET>                    ;ALL DONE
000636' 001000  000311
                                11400           ;DIVIDE FAC BY 10
                                11420           ;ALTERS A,B,C,D,E,H,L
000637' 001000  000315          11440   DIV10:  CALL    PUSHF           ;SAVE NUMBER
000640' 000000  001205'
000641' 000000  000625'
                                11460   IFN     LENGTH-2,<
000642' 001000  000301          11480           MOVRI   204,040,000,000 ;LOAD CONSTANT '10' INTO REGISTERS
000643' 000000  000240
000644' 000000  000204
000645' 001000  000021
000646' 000000  000000
000647' 000000  000000
000650' 001000  000315          11500           CALL    MOVFR>          ;MOVE THE CONSTANT TO THE FAC
000651' 000000  001229'
000652' 000000  000648'
                                11520   IFE     LENGTH-2,<
                                11540           LXI     H,FTEN          ;GET POINTER TO THE CONSTANT '10'
                                11560           CALL    MOVFM>          ;MOVE TEN INTO THE FAC
000653' 001000  000301          11580   FDIV1:  POPR                    ;GET NUMBER BACK IN REGISTERS
000654' 001000  000321
                                11600                                   ;FALL INTO DIVIDE AND WE ARE DONE
                                11660           ;DIVISION       FAC=ARG/FAC
                                11680           ;ALTERS A,B,C,D,E,H,L
000655' 001000  000357          11700   FDIV:   FSIGN                   ;CHECK FOR DIVISION BY ZERO
000656' 001000  000312          11720           JZ      DV0ERR          ;HE IS TRYING TO GET AWAY WITH IT
000657' 000000  000400
000660' 000000  000651'
000661' 001000  000056          11740           MVI     L,377           ;SUBTRACT THE TWO EXPONENTS, L IS A FLAG
000662' 000000  000377
000663' 001000  000315          11760           CALL    MULDIV          ;FIX UP THE EXPONENTS AND THINGS
000664' 000000  001035'
000665' 000000  000657'
000666' 001000  000064          11780           INR     M               ;ADD 2 TO EXPONENT TO CORRECT SCALING
000667' 001000  000064          11800           INR     M
                                11820           ;HERE WE SAVE THE FAC IN MEMORY SO WE CAN SUBTRACT IT FROM THE NUMBER
                                11840           ;IN THE REGISTERS QUICKLY.
```



```
000670' 001000  000053          11860           OCX     H               ;POINT TO MQ
000671' 001000  000176          11880           MOV     A,M             ;GET MQ
000672' 001000  000062          11900           STA     FDIV4+1         ;SAVE IT
000673' 000000  000734'
000674' 000000  000664'
000675' 001000  000053          11920           OCX     H               ;SAVE MIDDLE ORDER
000676' 001000  000176          11940           MOV     A,M
000677' 001000  000062          11960           STA     FDIV8+1         ;PUT IT WHERE NOTHING WILL HURT IT
000700' 000000  000730'
000701' 000000  000673'
000702' 001000  000053          11980           OCX     H               ;SAVE LO
000703' 001000  000176          12000           MOV     A,M
000704' 001000  000062          12020           STA     FDIVC+1
000705' 000000  000724'
000706' 000000  000700'
                                12040
                                12060   ;THE NUMERATOR WILL BE KEPT IN B,H,L.  THE QUOTIENT WILL BE FORMED IN C,D,E.
                                12080   ;TO GET A BIT OF THE QUOTIENT, WE FIRST SAVE B,H,L ON THE STACK, THEN
                                12100   ;SUBTRACT THE DENOMINATOR THAT WE SAVED IN MEMORY.  THE CARRY INDICATES
                                12120   ;WHETHER OR NOT B,H,L WAS BIGGER THAN THE DENOMINATOR.  IF B,H,L WAS BIGGER,
                                12140   ;THE NEXT BIT OF THE QUOTIENT IS A ONE.  TO GET THE OLD B,H,L OFF THE STACK,
                                12160   ;WE POP THEM INTO THE PSW.  IF THE DENOMINATOR WAS BIGGER, THE NEXT BIT OF
                                12180   ;THE QUOTIENT IS ZERO, AND WE GET THE OLD B,H,L BACK BY POPPING IT OFF THE
                                12200   ;STACK.  WE HAVE TO KEEP AN EXTRA BIT OF THE QUOTIENT IN FDIV8+1 IN CASE THE
                                12220   ;DENOMINATOR WAS BIGGER.  THEN B,H,L WILL GET SHIFTED LEFT.  IF THE MSB OF
                                12240   ;B WAS ONE, IT HAS TO BE STORED SOMEWHERE, SO WE STORE IT IN FDIV8+1.  THEN
                                12260   ;THE NEXT TIME THROUGH THE LOOP B,H,L WILL LOOK BIGGER BECAUSE IT HAS AN
                                12280   ;EXTRA MO BIT IN FDIV8+1.  WE ARE DONE DIVIDING WHEN THE MSB OF C IS A ONE.
                                12300   ;THIS OCCURS WHEN WE HAVE CALCULATED 24 BITS OF THE QUOTIENT.  WHEN WE JUMP
                                12320   ;TO ROUND, THE 25TH BIT OF THE QUOTIENT DETERMINES WHETHER WE ROUND OR NOT.
                                12340   ;IT IS IN THE MSB OF A.  IF INITIALLY THE DENOMINATOR IS BIGGER THAN THE
                                12360   ;NUMERATOR, THE FIRST BIT OF THE QUOTIENT WILL BE ZERO, THIS MEANS WE
                                12380   ;WILL GO THROUGH THE DIVIDE LOOP 26 TIMES, SINCE IT STOPS ON THE 25TH BIT
                                12400   ;AFTER THE FIRST NON-ZERO BIT OF THE EXPONENT.  SO, THIS QUOTIENT WILL LOOK
                                12420   ;SHIFTED LEFT ONE FROM THE QUOTIENT OF TWO NUMBERS IN WHICH THE NUMERATOR IS
                                12440   ;BIGGER.  THIS CAN ONLY occur ON THE FIRST TIME THROUGH THE LOOP.  SO C,D,E
                                12460   ;ARE ALL ZERO.  SO, IF WE FINISH THE LOOP AND C,D,E ARE ALL ZERO, THEN WE
                                12480   ;MUST DECREMENT THE EXPONENT TO CORRECT FOR THIS.
000707' 001000  000101          12500           MOV     B,C             ;GET NUMBER IN B,H,L
000710' 001000  000353          12520           XCHG
000711' 001000  000257          12540           XRA     A               ;ZERO C,D,E AND HIGHEST ORDER
000712' 001000  000117          12560           MOV     C,A
000713' 001000  000127          12580           MOV     D,A
000714' 001000  000137          12600           MOV     E,A
000715' 001000  000062          12620           STA     FDIV8+1
000716' 000000  000737'
000717' 000000  000725'
000720' 001000  000345          12640   FDIV1:  PUSH    H               ;SAVE LO'S OF NUMBER
000721' 001000  000305          12660           PUSH    B               ;SAVE HO OF NUMBER
000722' 001000  000175          12680           MOV     A,L             ;SUBTRACT NUMBER THAT WAS IN FAC
000723' 001000  000326          12700   FDIVC:  SUI     0               ;SUBTRACT LO
000724' 000000  000000
000725' 001000  000157          12720           MOV     L,A             ;SAVE IT
```

```
                                                13740           RAL
```



```
                                                13760           STA     FDIVG+1
                                                13780           ;CHECK SPECIAL CASES AND ADJ EXPONENTS FOR FMULT, FDIV
                                                13800           ;ALTERS A,B,H,L
                                                13820   IFE     LENGTH-2,<
                                                13840   MULDV$: MVI     A,377           ;ENTRY FROM DDIV, SUBTRACT EXPONENTS
                                                13860           XRD     1000,056        ;"MVI  L" AROUND NEXT BYTE
                                                13880   MULDVA: XRA     A               ;ENTRY FROM DMULT, ADD EXPONENTS
                                                13900           LXI     H,ARG-1         ;GET POINTER TO SIGN AND MO OF ARG
                                                13920           MOV     C,M             ;GET MO AND SIGN FOR UNPACKING
                                                13940           INX     H               ;INCREMENT POINTER TO EXPONENT
                                                13960           MOV     B,M             ;GET EXPONENT FOR BELOW
                                                13980           MOV     L,A>            ;SAVE ADD OR SUBTRACT FLAG
                                                14000   MULDIV: MOV     A,B             ;IS NUMBER IN REGISTERS ZERO?
                                                14020           ORA     A
                                                14040           JZ      MULDV2          ;IT IS, ZERO FAC AND WE ARE DONE
                                                14060
                                                14080           MOV     A,L             ;GET ADD OR SUBTRACT FLAG
                                                14100           LXI     H,FAC           ;GET POINTER TO EXPONENT
                                                14120
                                                14140           XRA     M               ;GET EXPONENT
                                                14160           ADD     B               ;ADD IN REGISTER EXPONENT
                                                14180           MOV     B,A             ;SAVE IT
                                                14200           RAR                     ;CHECK FOR OVERFLOW
                                                14220           XRA     B               ;OVERFLOW IF SIGN IS THE SAME AS LARRY
                                                14240           MOV     A,B             ;GET SUM
                                                14260           JP      MULDV1          ;WE HAVE OVERFLOW!!
                                                14280           ADI     200             ;PUT EXPONENT IN EXCESS 200
```

```
 985  001060' 000000  000000
 986  001061' 001000  000167          14300           MOV     M,A             ;SAVE IT IN THE FAC
 987  001062' 001000  000312          14320           JZ      POPRRT          ;WE HAVE UNDERFLOW, RETURN,
 988  001065' 000000  000230'
 989  001066' 000000  ww1055'
 990  001065' 001000  000315          14340           CALL    UNPACK          ;UNPACK THE ARGUMENTS
 991  001066' 000000  001272'
 992  001067' 000000  001263'
 993  001070' 001000  000167          14360           MOV     M,A             ;SAVE THE NEW SIGN
 994  001071' 001000  000053          14380           DCX     H               ;POINT TO EXPONENT
 995  001072' 001000  000311          14400           RET                     ;ALL DONE, LEAVE NO IN A
 996                                  14420   IFN     EXTFNC,<
 997  001073' 001000  000357          14440   MLOVEX1 FSIGN                   ;ENTRY FROM EXP, PICK UNDERFLOW IF NEGATIVE
 998  001074' 001000  002057          14460           CMA                     ;PICK OVERFLOW IF POSITIVE
 999  001075' 001000  000341          14480           POP     H>              ;DON'T SCREW UP STACK
1000  001076' 001000  000267          14500   MULOV1: ORA     A               ;IS ERROR OVERFLOW OR UNDERFLOW?
1001  001077' 001000  000341          14520   MULOV2: POP     H               ;GET OLD RETURN ADDRESS OFF STACK
1002                                  14540   IFE     LENGTH,<
1003                                  14560           JM      OVERR           ;OVERFLOW
1004                                  14580                                   ;UNDERFLOW => FALL INTO ZERO
1005                                  14600
1006                                  14620
1007                                  14640           ;ZERO FAC
1008                                  14660           ;ALTERS A ONLY
1009                                  14680           ;EXITS WITH A=0
1010                                  14700   ZERO:   XRA     A               ;ZERO A
1011                                  14720           STA     FAC             ;ZERO FAC
1012                                  14740           RET>                    ;ALL DONE
1013
1014
1015                                  14800   IFN     LENGTH,<
1016  001100' 001000  000362          14820           JP      ZERO            ;UNDERFLOW
1017  001101' 000000  000173'
1018  001102' 000000  001066'
1019  001103' 001000  000303          14840           JMP     OVERR>          ;OVERFLOW
1020  001104' 000000  000267'
1021  001105' 000000  001101'
1022
1023
1024                                  14900           ;MULTIPLY FAC BY 10
1025                                  14920           ;ALTERS A,B,C,D,E,H,L
1026  001106' 001000  000315          14940   MUL10:  CALL    MOVRF           ;GET NUMBER IN REGISTERS
1027  001107' 000000  001240'
1028  001110' 000000  001104'
1029  001111' 001000  000170          14960           MOV     A,B             ;GET EXPONENT
1030  001112' 001000  000267          14980           ORA     A               ;RESULT IS ZERO IF ARG IS ZERO
1031  001113' 00.000  000312          15000           RZ                      ;IT IS
1032  001114' 001000  000306          15020           ADI     2               ;MULTIPLY BY 4 BY ADDING 2 TO EXPONENT
1033  001115' 000000  000002
1034  001116' 001000  000332          15040           JC      OVERR           ;OVERFLOW!!
1035  001117' 000000  ww2267'
1036  001120' 000000  001107'
1037  001121' 001000  000107          15060           MOV     B,A             ;RESTORE EXPONENT
```



```
1038  001122' 001000  000315          15080           CALL    FADD            ;ADD IN ORIGINAL NUMBER TO GET 5 TIMES IT
1039  001123' 000000  ww0025'
1040  001124' 000000  001117'
1041  001125' 001000  000041          15100           LXI     H,FAC           ;ADD 1 TO EXPONENT TO MULTIPLY NUMBER BY
1042  001126' 000000  001044'
1043  001127' 000000  001123'
1044  001130' 001000  000064          15120           INR     M               ; 2 TO GET 10 TIMES ORIGINAL NUMBER
1045  001131' 001000  000300          15140           RNZ                     ;ALL DONE IF NO OVERFLOW
1046  001132' 001000  000303          15160           JMP     OVERR           ;OVERFLOW!!
1047  001133' 000000  000267'
1048  001134' 000000  001126'
1049                                  15180   PAGE
```

```
                                        13200  SUBTTL  SIGN, SGN, FLOAT, NEG AND ABS
                                        13210          ;PUT SIGN OF FAC IN A
                                        13220          ;ALTERS A ONLY
                                        13230          ;LEAVES FAC ALONE
                                        13240          ;NOTE: TO TAKE ADVANTAGE OF THE RST INSTRUCTIONS TO SAVE BYTES,
                                        13250          ;FSIGN IS DEFINED TO BE AN RST.  "FSIGN" IS EQUIVALENT TO "CALL SIGN"
                                        13260          ;THE FIRST FEW INSTRUCTIONS OF SIGN (THE ONES BEFORE SIGNC) ARE DONE
                                        13270          ;IN THE 8 BYTES AT THE RST LOCATION.
                                        13280  REPEAT  0,<
                                        13290  SIGN:   LDA     FAC             ;CHECK IF THE NUMBER IS ZERO
                                        13300          ORA     A
                                        13310          RZ>                     ;IT IS, A IS ZERO
0011351' 001000  000007*                13320  SIGNC:  LDA     FAC+1           ;GET SIGN OF FAC, IT IS NON-ZERO
0011361' 777777  777777*
0011371' 000000  001132'
                                        13330          XWD     1000,376        ;"CPI" AROUND NEXT BYTE
0011401' 001000  000057                 13340  FCOMPS: CMA                     ;ENTRY FROM FCOMP, COMPLEMENT SIGN
0011411' 001000  000027                 13350  ICOMPS: RAL                     ;ENTRY FROM ICOMP, PUT SIGN BIT IN CARRY
0011421' 001000  000237                 13360  SIGNS:  SBB     A               ;A=0 IF CARRY WAS 0, A=377 IF CARRY WAS 1
0011431' 001000  000300                 13370          RNZ                     ;RETURN IF NUMBER WAS NEGATIVE
0011441' 001000  000074                 13380  INRART: INR     A               ;PUT ONE IN A IF NUMBER WAS POSITIVE
0011451' 001000  000311                 13390          RET                     ;ALL DONE
                                        13400
                                        13410
                                        13420          ;SGN FUNCTION
                                        13430          ;ALTERS A,B,C,D,E,H,L
                                        13440  IFN     LENGTH=2,<
0011461' 001000  000357                 13450  SGN:    FSIGN>                  ;GET SIGN OF FAC IN A
                                        13460                                  ;FALL INTO FLOAT
                                        13470
                                        13480
                                        13490          ;FLOAT THE SIGNED INTEGER IN A
                                        13500          ;ALTERS A,B,C,D,E,H,L
0011501' 001000  000006                 13510  FLOAT:  MVI     B,210           ;SET EXPONENT CORRECTLY
0011511' 000000  000210
0011521' 001000  000021                 13520          LXI     D,$CODE         ;ZERO D,E
0011531' 000000  000601*
0011541' 000000  001136'
                                        13530                                  ;FALL INTO FLOATR
                                        13540
                                        13550
                                        13560          ;FLOAT THE SIGNED NUMBER IN B,A,D,E
                                        13570          ;ALTERS A,B,C,D,E,H,L
0011551' 001000  000041                 13580  FLOATR: LXI     H,FAC           ;GET POINTER TO FAC
0011561' 000000  001126*
0011571' 000000  001153'
0011601' 001000  000117                 13590          MOV     C,A             ;PUT MQ IN C
0011611' 001000  000160                 13600          MOV     M,B             ;PUT EXPONENT IN THE FAC
0011621' 001000  000026                 13610          MVI     B,0             ;ZERO OVERFLOW BYTE
0011631' 000000  000000
0011641' 001000  000043                 13620          INX     H               ;POINT TO SIGN
0011651' 001000  000066                 13630          MVI     M,200           ;ASSUME A POSITIVE NUMBER
0011661' 000000  000200
```



```
0011671' 001000  000027                 13640          RAL                     ;PUT SIGN IN CARRY
0011701' 001000  000303                 13650          JMP     FADFLT          ;GO AND FLOAT THE NUMBER
0011711' 000000  001143'
0011721' 000000  001156'
                                        13660
                                        13670
                                        13680          ;ABSOLUTE VALUE OF FAC
                                        13690          ;ALTERS A,H,L
0011731'                                13700  ABS:
                                        13710
0011731' 001000  000357                 13720  IFE     LENGTH=2,<
0011741' 001000  000360                 13730          CPI     2               ;IS THE ARGUMENT AN INTEGER?
                                        13740          JZ      IABS>           ;YES, USE THE INTEGER ABSOLUTE VALUE
                                        13750          FSIGN                   ;GET THE SIGN OF FAC
                                        13760          RP                      ;ALL DONE IF IT IS POSITIVE
                                        13770                                  ;FALL INTO NEG
                                        13780
                                        13790          ;NEGATE NUMBER IN THE FAC
                                        13800          ;ALTERS A,H,L
                                        13810          ;NOTE: THE NUMBER MUST BE PACKED
0011751' 001000  000041                 13820  NEG:    LXI     H,FAC+1         ;GET POINTER TO SIGN
0011761' 777777  777777*
0011771' 000000  001171'
0012001' 001000  000176                 13830          MOV     A,M             ;GET SIGN
0012011' 001000  000356                 13840          XRI     200             ;COMPLEMENT SIGN BIT
0012021' 000000  000200
0012041' 001000  000167                 13850          MOV     M,A             ;SAVE IT
0012051' 001000  000311                 13860          RET                     ;ALL DONE
                                        13870
                                        13880
                                        13890  IFE     LENGTH=2,<
                                        13900          ;NEGATE ANY TYPE VALUE IN THE FAC
                                        13910          ;ALTERS A,B,C,D,E,H,L
                                        13920  VNEG:   LDA     VALTYP          ;SEE WHAT KIND OF NUMBER WE HAVE
                                        13930          CPI     2
                                        13940          JZ      INEG            ;WE HAVE AN INTEGER, NEGATE IT THAT WAY
                                        13950          JM      TMERR           ;BLOW UP ON STRINGS
                                        13960          JMP     NEG             ;NEGATE SNG AND DBL THE SAME
                                        13970
                                        13980
                                        13990          ;SGN FUNCTION
                                        14000          ;ALTERS A,H,L
                                        14010  SGN:    CALL    VSIGN           ;GET THE SIGN OF THE FAC IN A
                                        14020          MOV     L,A             ;PUT IT IN THE LO POSITION
                                        14030          RAL                     ;EXTEND THE SIGN TO THE MQ
                                        14040          SBB     A
                                        14050          MOV     H,A
                                        14060          JMP     CONISS          ;RETURN THE RESULT AND SET VALTYP
                                        14070
                                        14080
                                        14090          ;GET THE SIGN OF THE VALUE IN THE FAC IN A
                                        14100          ;ASSUMES A HAS THE VALTYP WHEN CALLED
                                        14110          ;ALTERS A,H,L
```

```
                                17040   VSIGN:  CPI     2               ;IS THE ARGUMENT AN INTEGER?
                                17050           JNZ     SIGN            ;NO, SINGLE AND DOUBLE PREC, WORK THE SAME
                                17060           LHLD    FACLO           ;GET THE INTEGER ARGUMENT
                                17100           MOV     A,H             ;GET ITS SIGN
                                17120           ORA     L               ;CHECK IF THE NUMBER IS ZERO
                                17140           RZ                      ;IT IS, WE ARE DONE
                                17160           MOV     A,M             ;IT ISN'T, SIGN IS THE SIGN OF M
                                17180           JMP     ICOMPS>         ;GO SET A CORRECTLY
                                17200   PAGE
```



```
                                17220   SUBTTL  FLOATING POINT MOVEMENT ROUTINES
                                17240           ;PUT FAC ON STACK
                                17260           ;ALTERS D,E
001205' 001000  000353          17280   PUSHF:  XCHG                    ;SAVE (HL)
001206' 001052  000052          17300           LHLD    FACLO           ;GET LO'S
001207' 000000  001176'
                                17320           XTHL                    ;SWITCH LO'S AND RET ADDR
001211' 001000  000343          17340           PUSH    H               ;PUT RET ADDR BACK ON STACK
001212' 001000  000345          17360           LHLD    FAC+1           ;GET HO'S
001213' 001052  000052
001214' 777777  777777*
001215' 000000  001207'
                                17380           XTHL                    ;SWITCH HO'S AND RET ADDR
001216' 001000  000343          17400           PUSH    H               ;PUT RET ADDR BACK ON STACK
001217' 001000  000345          17420           XCHG                    ;GET OLD (HL) BACK
001220' 001000  000353          17440           RET                     ;ALL DONE
001221' 001000  000311
                                17460
                                17480
                                17500           ;MOVE NUMBER FROM MEMORY [(HL)] TO FAC
                                17520           ;ALTERS B,C,D,E,H,L
                                17540           ;AT EXIT NUMBER IS IN B,C,D,E
                                17560           ;AT EXIT (HL)1=(HL)+4
001222' 001000  000315          17580   MOVFM:  CALL    MOVRM           ;GET NUMBER IN REGISTERS
001223' 000000  001245'
001224' 000000  001214'
                                17600                                   ;FALL INTO MOVFR AND PUT IT IN FAC
                                17620
                                17640
                                17660           ;MOVE REGISTERS (B,C,D,E) TO FAC
                                17680           ;ALTERS D,E
001225' 001000  000353          17700   MOVFR:  XCHG                    ;GET LO'S IN (HL)
001226' 001000  000042          17720           SHLD    FACLO           ;PUT THEM WHERE THEY BELONG
001227' 000000  001207*
001230' 000000  001223'
001231' 001000  000140          17740           MOV     H,B             ;GET HO'S IN (HL)
001232' 001000  000151          17760           MOV     L,C
001233' 001000  000042          17780           SHLD    FAC+1           ;PUT HO'S WHERE THEY BELONG
001234' 777777  777777*
001235' 000000  001227'
001236' 001000  000353          17800           XCHG                    ;GET OLD (HL) BACK
001237' 001000  000311          17820           RET                     ;ALL DONE
                                17840
                                17860           ;MOVE FAC TO REGISTERS (B,C,D,E)
                                17900           ;ALTERS B,C,D,E,H,L
001240' 001000  000041          17920   MOVRF:  LXI     H,FACLO         ;GET POINTER TO FAC
001241' 000000  001227*
001242' 000000  001234'
                                17940                                   ;FALL INTO MOVRM
                                18000
                                18020           ;GET NUMBER IN REGISTERS (B,C,D,E) FROM MEMORY [(HL)]
                                18040           ;ALTERS B,C,D,E,H,L
```

```
                                        ;AT EXIT (HL)=(HL)+4
                        MOVRM:  MOV     E,M             ;GET LO
                                INX     H               ;POINT TO MO
                                MOV     D,M             ;GET MO
                                INX     H               ;POINT TO MO
                                MOV     C,M             ;GET MO
                                INX     H               ;POINT TO EXPONENT
                                MOV     B,M             ;GET EXPONENT
                        INXHRT: INX     H               ;INC POINTER TO BEGINNING OF NEXT NUMBER
                                RET                     ;ALL DONE


                                        ;MOVE NUMBER FROM FAC TO MEMORY ((HL))
                                        ;ALTERS A,B,D,E,H,L
                        MOVMF:  LXI     D,FACLO         ;GET POINTER TO FAC

                                        ;FALL INTO MOVE


                                        ;MOVE NUMBER FROM (DE) TO (HL)
                                        ;ALTERS A,B,D,E,H,L
                                        ;EXITS WITH (DE)=(DE)+4, (HL)=(HL)+4
                        MOVE:   MVI     B,4             ;SET COUNTER
                                IFE     LENGTH-2,<
                                XWD     1000,076        ;*MVI   A* OVER NEXT BYTE
                        MOVVFM: XCHG>                   ;MOVE NUMBERS INTO THE FAC
                        MOVE1:  LDAX    D               ;GET WORD, ENTRY FROM VMOVMF
                                MOV     M,A             ;PUT IT WHERE IT BELONGS
                                INX     D               ;INCREMENT POINTERS TO NEXT WORD
                                INX     H
                                DCR     B               ;SEE IF DONE
                                JNZ     MOVE1


                                RET


                                        ;UNPACK THE FAC AND THE REGISTERS
                                        ;ALTERS A,C,H,L
                                        ;WHEN THE NUMBER IN THE FAC IS UNACKED, THE ASSUMED ONE IN THE
                                        ;MANTISSA IS RESTORED, AND THE COMPLEMENT OF THE SIGN IS PLACED
                                        ;IN FAC+1
                        UNPACK: LXI     H,FAC+1         ;POINT TO MO AND SIGN

                                MOV     A,M             ;GET MO AND SIGN
                                PUSH    PSW             ;SAVE SIGN
                                ORI     200             ;RESTORE THE HIDDEN ONE


                                MOV     M,A             ;SAVE MO
                                POP     PSW             ;GET SIGN
```



```
                                XRA     M               ;GET COMPLEMENT OF SIGN IN MSB
                                INX     H               ;POINT TO TEMPORARY SIGN BYTE
                                INX     H
                                MOV     M,A             ;SAVE COMPLEMENT OF SIGN
                                MOV     A,C             ;GET MO AND SIGN OF THE REGISTERS
                                PUSH    PSW             ;SAVE SIGN
                                ORI     200             ;RESTORE THE HIDDEN ONE


                                MOV     C,A             ;SAVE THE MO
                                POP     PSW             ;GET THE SIGN BACK
                                XRA     M               ;COMPARE SIGN OF FAC AND SIGN OF REGISTERS
                                RET                     ;ALL DONE


                        IFE     LENGTH-2,<
                        REPEAT  8,<                     ;VPUSHF WILL BE IN-LINE IN F3
                                ;PUT ANY TYPE VALUE ON THE STACK FROM THE FAC
                                ;STRINGS ARE TREATED AS INTEGERS
                                ;ALTERS A,B,C,H,L
                        VPUSHF: LDA     VALTYP          ;GET THE VALUE TYPE
                                CPI     4               ;SET FLAGS ACCORDING TO VALTYP
                                LXI     H,FACLO         ;GET POINTER TO LO IN FAC
                                PUSHM                   ;PUSH FACLO+0,1 ON THE STACK
                                JM      VPUSHD          ;RETURN IF THE DATA WAS AN INTEGER OR A STRING
                                PUSHM                   ;PUSH FAC+1,0 ON THE STACK
                                JZ      VPUSHD          ;RETURN IF WE HAD A SINGLE PRECISION NUMBER
                                LXI     D,DFACLO        ;WE HAVE A DOUBLE PRECISON NUMBER
                                PUSHM                   ;PUSH IT'S 4 LO BYTES ON THE STACK
                                PUSHM
                        VPUSHD:>                        ;ALL DONE


                                ;MOVE ANY TYPE VALUE FROM MEMORY ((HL)) TO FAC
                                ;ALTERS A,B,D,E,H,L
                        VMOVFA: LXI     H,ARGLO         ;ENTRY FROM DADD, MOVE ARG TO FAC
                        VMOVFM: LXI     D,MOVVFM        ;GET ADDRESS OF LOCATION THAT DOES
                                JMP     VMVVFM          ; AN "XCHG" AND FALLS INTO MOVE1


                                ;MOVE ANY TYPE VALUE FROM FAC TO MEMORY ((HL))
                                ;ALTERS A,B,D,E,H,L
                        VMOVAF: LXI     H,ARGLO         ;ENTRY FROM FIN, OMUL10, DDIV10
                                ;MOVE FAC TO ARG
                        VMOVMF: LXI     D,MOVE1         ;GET ADDRESS OF MOVE SUBROUTINE
                        VMVVFM: PUSH    D               ;MOVE IT ON THE STACK
                                LXI     D,FACLO         ;GET FIRST ADDRESS FOR INT, SNG
                                LDA     VALTYP          ;GET THE VALUE TYPE
                                ANI     177             ;STRINGS LOOK LIKE REALS
                                MOV     B,A             ;SET UP THE COUNT
                                CPI     10              ;DO WE HAVE DBL?
                                RNZ                     ;WE DO NOT, GO DO THE MOVE
                                LXI     D,DFACLO        ;WE DO, GET LO ADDR OF THE DBL NUMBER
                                RET>                    ;GO DO THE MOVE
```

1324   19980   PAGE


```
1325                                    20002   SUBTTL  COMPARE TWO NUMBERS
1326                                    20022           ;COMPARE TWO SINGLE PRECISION NUMBERS
1327                                    20042           ;A=1 IF ARG .LT. FAC
1328                                    20062           ;A=0 IF ARG=FAC
1329                                    20082           ;A=-1 IF ARG .GT. FAC
1330                                    20102           ;DOREL DEPENDS UPON THE FACT THAT FCOMP RETURNS WITH CARRY ON
1331                                    20122           ; IFF A HAS 377
1332                                    20142           ;ALTERS A,H,L
1333  001317' 001000  000170            20162   FCOMP:  MOV     A,B             ;CHECK IF ARG IS ZERO
1334  001320' 001000  000267            20182           ORA     A
1335  001321' 001000  000312            20202           JZ      SIGN
1336  001324' 000000  000000*
1337  001325' 000000  001273'
1338  001326' 001000  000041            20222           LXI     H,FCOMPS        ;NE JUMP TO FCOMPS WHEN WE ARE DONE
1339  001329' 000000  001141'
1340  001326' 000000  001322'
1341  001327' 001000  000345            20242           PUSH    H               ;PUT THE ADDRESS ON THE STACK
1342  001330' 001000  000357            20262           FSIGN                   ;CHECK IF FAC IS ZERO
1343  001331' 001000  000171            20282           MOV     A,C             ;IF IT IS, RESULT IS MINUS THE SIGN OF ARG
1344  001332' 001000  000310            20302           RZ                      ;IT IS
1345  001333' 001000  000041            20322           LXI     H,FAC+1         ;POINT TO SIGN OF FAC
1346  001334' 777777  777777*
1347  001335' 000000  001325'
1348  001336' 001000  000256            20342           XRA     M               ;SEE IF THE SIGNS ARE THE SAME
1349  001337' 001000  000171            20362           MOV     A,C             ;IF THEY ARE DIFFERENT, RESULT IS SIGN OF ARG
1350  001340' 001000  000370            20382           RM                      ;THEY ARE DIFFERENT
1351  001341' 001000  001315            20402           CALL    FCOMP2          ;CHECK THE REST OF THE NUMBER
1352  001342' 000000  001347'
1353  001343' 000000  001534'
1354  001344' 001000  000037            20422   FCOMPO: RAR                     ;NUMBERS ARE DIFFERENT, CHANGE SIGN IF
1355  001345' 001000  000251            20442           XRA     C               ; BOTH NUMBERS ARE NEGATIVE
1356  001346' 001000  000311            20462           RET                     ;GO SET UP A
1357
1358  001347' 001000  000043            20502   FCOMP2: INX     H               ;POINT TO EXPONENT
1359  001350' 001000  000170            20522           MOV     A,B             ;GET EXPONENT OF ARG
1360  001351' 001000  000276            20542           CMP     M               ;COMPARE THE TWO
1361  001352' 001000  000300            20562           RNZ                     ;NUMBERS ARE DIFFERENT
1362  001353' 001000  000053            20582           DCX     H               ;POINT TO MD
1363  001354' 001000  000171            20602           MOV     A,C             ;GET MD OF ARG
1364  001355' 001000  000276            20622           CMP     M               ;COMPARE WITH MD OF FAC
1365  001356' 001000  000300            20642           RNZ                     ;THEY ARE DIFFERENT
1366  001357' 001000  000053            20662           DCX     H               ;POINT TO MD OF FAC
1367  001360' 001000  000172            20682           MOV     A,D             ;GET MD OF ARG
1368  001361' 001000  000276            20702           CMP     M               ;COMPARE WITH MD OF FAC
1369  001362' 001000  000500            20722           RNZ                     ;THE NUMBERS ARE DIFFERENT
1370  001363' 001000  000053            20742           DCX     H               ;POINT TO LO OF FAC
1371  001364' 001000  000173            20762           MOV     A,E             ;GET LO OF ARG
1372  001365' 001000  000226            21002           SUB     M               ;SUBTRACT LO OF FAC
1373  001366' 001000  000300            21022           RNZ                     ;NUMBERS ARE DIFFERENT
1374  001367' 001000  000341            21042           POP     H               ;NUMBERS ARE THE SAME, DON'T SCREW UP STACK
1375  001370' 001000  000341            21062           POP     H
1376  001371' 001000  000311            21082           RET                     ;ALL DONE
1377
```

```
                                20920   IFE     LENGTH-2,<
                                20940           ;COMPARE TWO INTEGERS
                                20960           ;A=1 IF (DE) .LT. (HL)
                                20980           ;A=0 IF (DE)=(HL)
                                21000           ;A=-1 IF (DE) .GT. (HL)
                                21020           ;ALTERS A ONLY
                                21040   ICOMP:  MOV     A,D             ;ARE THE SIGNS THE SAME?
                                21060           XRA     H
                                21080           MOV     A,H             ;IF NOT, ANSWER IS THE SIGN OF (HL)
                                21100           JM      ICOMPS          ;THEY ARE DIFFERENT
                                21120           CMP     D               ;THEY ARE THE SAME, COMPARE THE HO'S
                                21140           JNZ     SIGNS           ;GO SET UP A
                                21160           MOV     A,L             ;COMPARE THE LO'S
                                21180           SUB     E
                                21200           JNZ     SIGNS           ;GO SET UP A
                                21220           RET                     ;ALL DONE, THEY ARE THE SAME
                                21240
                                21260
                                21280           ;COMPARE TWO DOUBLE PRECISION NUMBERS
                                21300           ;A=1 IF ARG .LT. FAC
                                21320           ;A=0 IF ARG=FAC
                                21340           ;A=-1 IF ARG .GT. FAC
                                21360           ;ALTERS A,B,C,D,E,H,L
                                21380   DCOMPJ: LXI     H,ARGLO         ;ENTRY WITH POINTER TO ARG IN (DE)
                                21400           MVI     B,10            ;SET UP COUNT TO MOVE DBL NUMBERS
                                21420           CALL    MOVEI           ;MOVE THE ARGUMENT INTO ARG
                                21440   DCOMP:  LXI     D,ARG           ;GET POINTER TO ARG
                                21460           LDAX    D               ;SEE IF ARG=0
                                21480           ORA     A
                                21500           JZ      SIGN            ;ARG=0, GO SET UP A
                                21520           LXI     H,FCOMPS        ;PUSH FCOMPS ON STACK SO WE WILL RETURN TO
                                21540           PUSH    H               ; TO IT AND SET UP A
                                21560           FSIGN                   ;SEE IF FAC=0
                                21580           DCX     D               ;POINT TO SIGN OF ARG
                                21600           LDAX    D               ;GET SIGN OF ARG
                                21620           MOV     C,A             ;SAVE IT FOR LATER
                                21640           RZ                      ;FAC=0, SIGN OF RESULT IS SIGN OF ARG
                                21660           LXI     H,FAC+1         ;POINT TO SIGN OF FAC
                                21680           XRA     M               ;SEE IF THE SIGNS ARE THE SAME
                                21700           MOV     A,C             ;IF THEY ARE, GET THE SIGN OF THE NUMBERS
                                21720           RM                      ;THE SIGNS ARE DIFFERENT, GO SET A
                                21740           INX     D               ;POINT BACK TO EXPONENT OF ARG
                                21760           INX     H               ;POINT TO EXPONENT OF FAC
                                21780           MVI     B,10            ;SET UP A COUNT
                                21800   DCOMP1: LDAX    D               ;GET A BYTE FROM ARG
                                21820           SUB     M               ;COMPARE IT WITH THE FAC
                                21840           JNZ     FCOMPO          ;THEY ARE DIFFERENT, GO SET UP A
                                21860           DCX     D               ;THEY ARE THE SAME, EXAMINE THE NEXT LOWER
                                21880           DCX     H               ; ORDER BYTES
                                21900           DCR     B               ;ARE WE DONE?
                                21920           JNZ     DCOMP1          ;NO, COMPARE THE NEXT BYTES
                                21940           POP     B               ;THEY ARE THE SAME, GET FCOMPS OFF STACK
```



```
                                21960           RET>                    ;ALL DONE
                                21980   PAGE
```

```
                        22000 $SUBTTL  CONVERSION ROUTINES BETWEEN INTEGER, SINGLE AND DOUBLE PRECISION
                        22020 IFE      LENGTH-2,<
                        22040         ;FORCE THE FAC TO BE AN INTEGER
                        22060         ;ALTERS A,B,C,D,E,M,L
                        22080 FRCINT: LDA     VALTYP            ;SEE WHAT WE HAVE
                        22100         CPI     4
                        22120         LHLD    FACLO             ;GET FACLO-B,1 IN CASE WE HAVE AN INTEGER
                        22140         RC                        ;WE HAVE AN INTEGER, ALL DONE
                        22160         JM      TMERR             ;WE HAVE A STRING, THAT IS A "NO-NO"
                        22180         CNZ     CONSD             ;IF WE HAVE A DBL, CONVERT IT TO A SNG
                        22200         LXI     H,OVERR           ;PUT OVERR ON THE STACK SO WE WILL GET ERROR
                        22220         PUSH    H                 ; IF NUMBER IS TOO BIG
                        22240                                   ;FALL INTO CONIS
                        22260
                        22280
                        22300         ;CONVERT SINGLE PRECISION NUMBER TO INTEGER
                        22320         ;ALTERS A,B,C,D,E,M,L
                        22340 CONIS:  LDA     FAC               ;GET THE EXPONENT
                        22360         CPI     220               ;SEE IF IT IS TOO BIG
                        22380         JNC     CONIS2            ;IT IS, BUT IT MIGHT BE -32768
                        22400         CALL    QINT              ;IT ISN'T, CONVERT IT TO AN INTEGER
                        22420         XCHG                      ;PUT IT IN (HL)
                        22440 CONIS1: POP     D                 ;GET ERROR ADDRESS OFF STACK
                        22460         ;ENTRY FROM SGN, FIN, LINPRI
                        22480 CONIS8: SHLD    FACLO             ;STORE THE NUMBER IN FACLO
                        22500         MVI     A,2               ;SET VALTYP TO "INTEGER"
                        22520 CONIS0: STA     VALTYP            ;ENTRY FROM CONDS
                        22540         RET                       ;ALL DONE
                        22560 CONIS2: MOVRI   220,200,000,000 ;CHECK IF NUMBER IS -32768, ENTRY FROM FIN
                        22580         CALL    FLOMP
                        22600         RNZ                       ;ERROR:  IT CAN'T BE CONVERTED TO AN INTEGER
                        22620         MOV     H,C               ;IT IS -32768, PUT IT IN (HL)
                        22640         MOV     L,D
                        22660         JMP     CONIS1            ;STORE IT IN THE FAC AND SET VALTYP
                        22680
                        22700
                        22720         ;FORCE THE FAC TO BE A SINGLE PRECISION NUMBER
                        22740         ;ALTERS A,B,C,D,E,M,L
                        22760 FRCSNG: LDA     VALTYP            ;SEE WHAT KIND OF NUMBER WE HAVE
                        22780         CPI     4
                        22800         RZ                        ;WE ALREADY HAVE AN INTEGER, ALL DONE
                        22820         JC      CONSI             ;WE HAVE AN INTEGER, CONVERT IT
                        22840         JM      TMERR             ;STRINGS!! -- ERROR!!
                        22860                                   ;DBL PREC -- FALL INTO CONSD
                        22880
                        22900
                        22920         ;CONVERT DOUBLE PRECISION NUMBER TO A SINGLE PRECISION ONE
                        22940         ;ALTERS A,B,C,D,E,M,L
                        22960 CONSD:  CALL    MOVRF             ;GET THE MOF'S IN THE REGISTERS
                        22980         MVI     A,4               ;SET VALTYP TO "SINGLE PRECISON"
                        23000         STA     VALTYP
                        23020         MOV     A,B               ;CHECK IF THE NUMBER IS ZERO
                        23040         ORA     A
```



```
                        23060         RZ                        ;IF IT IS, WE ARE DONE
                        23080         CALL    UNPACK            ;UNPACK THE NUMBER
                        23100         LXI     H,FACLO-1         ;GET FIRST BYTE BELOW A SNG NUMBER
                        23120         MOV     B,M               ;PUT IT IN B FOR ROUND
                        23140         JMP     ROUND             ;ROUND THE DBL NUMBER UP AND WE ARE DONE
                        23160
                        23180
                        23200         ;CONVERT AN INTEGER TO A SINGLE PRECISION NUMBER
                        23220         ;ALTERS A,B,C,D,E,M,L
                        23240 CONSI:  LHLD    FACLO             ;GET THE INTEGER
                        23260 CONSIN: MVI     A,4               ;SET VALTYP TO "SINGLE PRECISION"
                        23280         STA     VALTYP
                        23300         MOV     A,M               ;SET UP REGISTERS FOR FLOATH
                        23320         MOV     C,L
                        23340         MVI     E,0
                        23360         MVI     B,220
                        23380         JMP     FLOATR            ;GO FLOAT THE NUMBER
                        23400
                        23420
                        23440         ;FORCE THE FAC TO BE A DOUBLE PRECISION NUMBER
                        23460         ;ALTERS A,B,C,D,E,M,L
                        23480 FRCDBL: LDA     VALTYP            ;SEE WHAT KIND OF NUMBER WE HAVE
                        23500         CPI     10
                        23520         RZ                        ;WE ALREADY HAVE A DBL, WE ARE DONE
                        23540         JNC     TMERR             ;GIVE AN ERROR IF WE HAVE A STRING
                        23560         CPI     2                 ;SEE IF WE HAVE A SNG OR INT
                        23580         CZ      CONSI             ;CONVERT TO SNG IF WE HAVE AN INT
                        23600                                   ;FALL INTO CONDS AND CONVERT TO DBL
                        23620
                        23640
                        23660         ;CONVERT A SINGLE PRECISION NUMBER TO A DOUBLE PRECISION ONE
                        23680         ;ALTERS A,H,L
                        23700 CONDS:  LXI     H,$CODE
                        23720         SHLD    DFACLO            ;CLEAR THE FOUR LOWER BYTES IN THE DOUBLE
                        23740         SHLD    DFACLO+2          ; PRECISION NUMBER
                        23760         MVI     A,10              ;SET VALTYP TO "DOUBLE PRECISION"
                        23780         JMP     CONIS0            ;GO TO IT
                        23800 PAGE
```

```
                                        23820  SUBTTL  GREATEST INTEGER FUNCTION
                                        23840          ;QUICK GREATEST INTEGER FUNCTION
                                        23860          ;LEAVES INT(FAC) IN C,D,E (SIGNED)
                                        23880          ;ASSUMES FAC .LT. 2**23 = 8388608
                                        23900          ;ASSUMES THE EXPONENT OF FAC IS IN A
                                        23920          ;ALTERS A,B,C,D,E
1530  001372' 001000  000107           23940  QINT:   MOV     B,A             ;ZERO B,C,D,E IN CASE THE NUMBER IS ZERO
1531  001373' 001000  000117           23960          MOV     C,A
1532  001374' 001000  000127           23980          MOV     D,A
1533  001375' 001000  000137           24000          MOV     E,A
1534  001376' 001000  000267           24020          ORA     A               ;SET CONDITION CODES
1535  001377' 001000  000310           24040          RZ                      ;IT IS ZERO, WE ARE DONE
1536
1537                                    24060  ;THE HARD CASE IN QINT IS NEGATIVE NON-INTEGERS.  TO HANDLE THIS, IF THE
1538                                    24080  ;NUMBER IS NEGATIVE, WE REGARD THE 3-BYTE MANTISSA AS A 3-BYTE INTEGER AND
1539                                    24100  ;SUBTRACT ONE.  THEN ALL THE FRACTIONAL BITS ARE SHIFTED OUT BY SHIFTING THE
1540                                    24120  ;MANTISSA RIGHT.  THEN, IF THE NUMBER WAS NEGATIVE, WE ADD ONE.  SO, IF WE
1541                                    24140  ;HAD A NEGATIVE INTEGER, ALL THE BITS TO THE RIGHT OF THE BINARY POINT WERE
1542                                    24160  ;ZERO.  SO THE NET EFFECT IS WE HAVE THE ORIGINAL NUMBER IN C;D;E.  IF THE
1543                                    24180  ;NUMBER WAS A NEGATIVE NON-INTEGER, THERE IS AT LEAST ONE NON-ZERO BIT TO THE
1544                                    24200  ;RIGHT OF THE BINARY POINT.  SO THE NET EFFECT IS THAT WE GET THE ABSOLUTE
1545                                    24220  ;VALUE OF INT(FAC) IN C,D,E.  C,D,E IS THEN NEGATED IF THE ORIGINAL NUMBER WAS
1546                                    24240  ;NEGATIVE SO THE RESULT WILL BE SIGNED.
1547  001400' 001000  000345           24260          PUSH    H               ;SAVE (HL)
1548  001401' 001000  000315           24280          CALL    MOVRF           ;GET NUMBER IN THE REGISTERS
1549  001402' 000000  001240'
1550  001405' 000000  001342'
1551  001404' 001000  000315           24320          CALL    UNPACK          ;UNPACK THE NUMBER
1552  001405' 000000  001272'
1553  001406' 000000  001406'
1554  001407' 001000  000256           24340          XRA     M               ;GET SIGN OF NUMBER
1555  001410' 001000  000167           24360          MOV     M,A             ;DON'T LOSE IT
1556  001411' 001000  000374           24380          CM      QINTA           ;SUBTRACT 1 FROM LO IF NUMBER IS NEGATIVE
1557  001412' 000000  001436'
1558  001413' 000000  001405'
1559  001414' 001000  003276           24400          MVI     A,230           ;SEE HOW MANY WE HAVE TO SHIFT TO CHANGE
1560  001415' 000000  000230
1561  001416' 001000  000220           24420          SUB     B               ; NUMBER TO AN INTEGER
1562  001417' 001000  000315           24440          CALL    SHIFTR          ;SHIFT NUMBER TO GET RID OF FRACTIONAL BITS
1563  001420' 000000  000334'
1564  001421' 000000  001412'
1565  001422' 001000  000174           24460          MOV     A,M             ;GET SIGN
1566  001423' 001000  000027           24480          RAL                     ;PUT SIGN IN CARRY SO IT WILL NOT BE CHANGED
1567  001424' 001000  000334           24500          CC      ROUNDA          ;IF NUMBER WAS NEGATIVE, ADD ONE
1568  001425' 000000  000C55'
1569  001426' 000000  001420'
1570  001427' 001000  000006           24520          MVI     B,0             ;FORGET THE BITS WE SHIFTED OUT
1571  001430' 000000  000000
1572  001431' 001000  000334           24540          CC      NEGR            ;NEGATE NUMBER IF IT WAS NEGATIVE BECAUSE WE
1573  001432' 000000  000310'                          ;
1574  001433' 000000  001425'                          ;
1575                                    24560                                  ; WANT A SIGNED MANTISSA
1576  001434' 001000  000341           24580          POP     H               ;GET OLD (HL) BACK
```



```
1577  001435' 001000  000311           24600          RET                     ;ALL DONE
1578
1579  001436' 001000  000033           24640  QINTA:  DCX     D               ;SUBTRACT ONE FROM C,D,E
1580  001437' 001000  000172           24660          MOV     A,D             ;WE HAVE TO SUBTRACT ONE FROM C IF
1581  001440' 001000  000243           24680          ANA     E               ; D AND E ARE BOTH ALL ONES
1582  001441' 001000  000074           24700          INR     A               ;SEE IF BOTH WERE -1
1583  001442' 001000  000300           24720          RNZ                     ;THEY WERE NOT, WE ARE DONE
1584                                    24740  IFN     LENGTH-2,<
1585  001443' 001000  000015           24760          DCR     C>              ;THEY WERE, SUBTRACT ONE FROM C
1586                                    24780  IFE     LENGTH-2,<
1587                                    24800  DCX8RT: DCX     B>              ;THIS IS FOR BILL.  C WILL NEVER BE ZERO
1588                                    24820                                  ; (THE MSB WILL ALWAYS BE ONE) SO "DCX  B"
1589                                    24840                                  ; AND "DCR      C" ARE FUNCTIONALLY EQUIVALENT
1590  001444' 001000  000311           24860          RET                     ;ALL DONE
1591
1592
1593                                    24920          ;GREATEST INTEGER FUNCTION
1594                                    24940          ;ALTERS A,B,C,D,E,H,L
1595                                    24960  IFE     LENGTH-2,<
1596                                    24980  INTFNC: CPI     4               ;SEE WHAT KIND OF NUMBER WE HAVE
1597                                    25000          RC                      ;IT IS AN INTEGER, ALL DONE
1598                                    25020          JNZ     DINT            ;CONVERT THE DOUBLE PRECISION NUMBER
1599                                    25040          CALL    CONIS>          ;TRY TO CONVERT THE NUMBER TO AN INTEGER
1600                                    25060                                  ;IF WE CAN'T, WE WILL RETURN HERE TO GIVE A
1601                                    25080                                  ; SINGLE PRECISION RESULT
1602  001445' 001002  000041           25100          LXI     H,FAC           ;GET EXPONENT
1603  001446' 000002  001156*
1604  001447' 000002  061432'
1605  001450' 001000  000176           25120          MOV     A,M
1606  001451' 001000  003270           25140          CPI     230             ;SEE IF NUMBER HAS ANY FRACTIONAL BITS
1607  001452' 000000  000230
1608                                    25160  IFN     EXTFNC,<               ;THE ONLY GUY WHO NEEDS THIS DOESN'T CARE
1609  001453' 001000  000072           25180          LDA     FACLO>          ; ABOUT THE SIGN
1610  001454' 000000  001253*
1611  001455' 001000  001446'
1612  001456' 001000  000320           25200          RNC                     ;IT DOES NOT
1613                                    25220  IFN     EXTFNC,<
1614  001457' 001000  000176           25240          MOV     A,M>            ;GET EXPONENT BACK
1615  001460' 001000  000315           25260          CALL    QINT            ;IT DOES, SHIFT THEM OUT
1616  001461' 000000  001372'
1617  001462' 000002  001454'
1618  001463' 001000  000066           25280          MVI     M,230           ;CHANGE EXPONENT SO IT WILL BE CORRECT
1619  001464' 000000  000230
1620                                    25300                                  ; AFTER NORMALIZATION
1621                                    25320  IFN     EXTFNC,<
1622  001465' 001000  000173           25340          MOV     A,E             ;GET LO
1623  001466' 001000  000365           25360          PUSH    PSW>            ;SAVE IT
1624  001467' 001000  000171           25380          MOV     A,C             ;NEGATE NUMBER IF IT IS NEGATIVE
1625  001470' 001000  000027           25400          RAL                     ;PUT SIGN IN CARRY
1626                                    25420  IFE     EXTFNC,<
1627                                    25440          JMP     FADFLT>         ;REFLOAT NUMBER
1628                                    25460  IFN     EXTFNC,<
1629  001471' 001000  000315           25480          CALL    FADFLT          ;REFLOAT NUMBER
```

```
1630  001472' 000020  001437'
1631  001475' 000020  001461'
1632  001478' 001000  000361          25500   POPPRT: POP     PSW             ;GET LO BACK
1633  001479' 001000  000311          25520           RET>                    ;ALL DONE
1634
1635
1636                                   25580   IFE     LENGTH=2,<
1637                                   25620           ;GREATEST INTEGER FUNCTION FOR DOUBLE PRECISION NUMBERS
1638                                   25640           ;ALTERS A,B,C,D,E,H,L
1639                                   25660   DINT:   LXI     H,FAC           ;GET POINTER TO FAC
1640                                   25680           MOV     A,M             ;GET EXPONENT
1641                                   25700           CPI     220             ;CAN WE CONVERT IT TO AN INTEGER?
1642                                   25720           JC      FRCINT          ;THEN DO SO
1643                                   25740           JNZ     CINT2           ;CHECK FOR -32768
1644                                   25760           MOV     C,A             ;SAVE EXPONENT IN C
1645                                   25780           DCX     H               ;GET POINTER TO SIGN AND MO
1646                                   25800           MOV     A,M             ;GET SIGN AND MO
1647                                   25820           XRI     200             ;CHECK IF IT IS 200
1648                                   25840           MVI     B,6             ;SET UP A COUNT TO CHECK IF THE REST OF
1649                                   25860   CINT1:  DCX     H               ; THE NUMBER IS ZERO, POINT TO NEXT BYTE
1650                                   25880           ORA     M               ;IF ANY BITS ARE NON-ZERO, A WILL BE NON-ZERO
1651                                   25900           DCR     B               ;ARE WE DONE?
1652                                   25920           JNZ     CINT1           ;NO, CHECK THE NEXT LOWER ORDER BYTE
1653                                   25940           ORA     A               ;IS A NOW ZERO?
1654                                   25960           LXI     H,200A400+3CODE ;GET -32768 JUST IN CASE
1655                                   25980           JZ      CON1SS          ;A IS ZERO SO WE HAVE -32768
1656                                   26000           MOV     A,C             ;GET EXPONENT
1657                                   26020   CINT2:  CPI     270             ;ARE THERE ANY FRACTIONAL BITS?
1658                                   26040           RNC                     ;NO, THE NUMBER IS ALREADY AN INTEGER
1659                                   26060   CINTFO: PUSH    PSW             ;ENTRY FROM FOUT, CARRY IS ZERO IF WE COME
1660                                   26080           ; HERE FROM FOUT
1661                                   26100           CALL    MOVRF           ;GET MO'S OF NUMBER IN REGISTERS FOR UNPACKING
1662                                   26120           CALL    UNPACK          ;UNPACK IT
1663                                   26140           XRA     M               ;GET ITS SIGN BACK
1664                                   26160           DCX     H               ;SET THE EXPONENT TO NORMALIZE CORRECTLY
1665                                   26180           MVI     M,270
1666                                   26200           PUSH    PSW             ;SAVE THE SIGN
1667                                   26220           CM      DINTA           ;SUBTRACT 1 FROM LO IF NUMBER IS NEGATIVE
1668                                   26240           MVI     A,270           ;GET HOW MANY BITS WE HAVE TO SHIFT OUT
1669                                   26260           SUB     B
1670                                   26280           CALL    DSHFTR          ;SHIFT THEM OUT,!
1671                                   26300           POP     PSW             ;GET THE SIGN BACK
1672                                   26320           CM      DROUNA          ;IF NUMBER WAS NEGATIVE, ADD ONE
1673                                   26340           XRA     A               ;PUT A ZERO IN THE EXTRA LO BYTE SO WHEN
1674                                   26360           STA     DFACLO-1        ; WE NORMALIZE, WE WILL SHIFT IN ZEROS
1675                                   26380           POP     PSW             ;IF WE WERE CALLED FROM FOUT, DON'T NORMALIZE,
1676                                   26380           RNC                     ; JUST RETURN
1677                                   26400           JMP     DNORML          ;RE-FLOAT THE INTEGER
1678                                   26420
1679                                   26440   DINTA:  LXI     H,DFACLO        ;SUBTRACT ONE FROM FAC, GET POINTER TO LO
1680                                   26460   DINTA1: MOV     A,M             ;GET A BYTE OF FAC
1681                                   26480           DCR     M               ;SUBTRACT ONE FROM IT
1682                                   26500           ORA     A               ;CONTINUE ONLY IF THE BYTE USED TO BE ZERO
```



```
1683                                   26520           INX     H               ;INCREMENT POINTER TO NEXT BYTE
1684                                   26540           JZ      DINTA1          ;CONTINUE IF NECESSARY
1685                                   26560           RET>                    ;ALL DONE
1686                                   26580   PAGE
```

```
MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF MACRO 47(113) 06109 27-AUG-75 PAGE 10
F8      MAC     23-AUG-64 06108         INTEGER ARITHMETIC ROUTINES

                                        ;INTEGER SUBTRACTION    (ML)=(DE)-(ML)
                                        ;ALTERS A,B,C,D,E,H,L
                        ISUB:   MOV     A,H             ;EXTEND THE SIGN OF (ML) TO B
                                RAL                     ;GET SIGN IN CARRY
                                SBB     A
                                MOV     B,A
                                CALL    INEGHL          ;NEGATE (ML)
                                MOV     A,C             ;GET A ZERO
                                SUB     B               ;NEGATE SIGN
                                JMP     IADDS           ;GO ADD THE NUMBERS

                                ;INTEGER ADDITION       (HL)=(DE)+(HL)
                                ;ALTERS A,B,C,D,E,H,L
                        IADD:   MOV     A,H             ;EXTEND THE SIGN OF (ML) TO B
                                RAL                     ;GET SIGN IN CARRY
                                SBB     A
                        IADDS:  MOV     B,A             ;SAVE THE SIGN
                                PUSH    H               ;SAVE THE SECOND ARGUMENT IN CASE OF OVERFLOW
                                MOV     A,D             ;EXTEND THE SIGN OF (DE) TO A
                                RAL                     ;GET SIGN IN CARRY
                                SBB     A
                                DAD     D               ;ADD THE TWO LO'S
                                ADC     B               ;ADD THE EXTRA HO
                                RRC                     ;IF THE LSB OF A IS DIFFERENT FROM THE MSB OF
                                XRA     H               ; H, THEN OVERFLOW OCCURED
                                JP      POPRT           ;NO OVERFLOW, GET OLD (ML) OFF STACK AND WE
                                                        ; ARE DONE
                                PUSH    B               ;OVERFLOW -- SAVE EXTENDED SIGN OF (ML)
                                XCHG                    ;GET (DE) IN (ML)
                                CALL    CONSIM          ;FLOAT IT
                                POP     PSW             ;GET SIGN OF (ML) IN A
                                POP     H               ;GET OLD (ML) BACK
                                CALL    PUSHF           ;PUT FIRST ARGUMENT ON STACK
                                XCHG                    ;PUT SECOND ARGUMENT IN (DE) FOR FLOATR
                                CALL    INEGAD          ;FLOAT IT
                                POPR                    ;GET FIRST ARGUMENT OFF STACK
                                JMP     FADD            ;ADD THE TWO NUMBERS USING SINGLE PRECISION

                                ;INTEGER MULTIPLICATION         (HL)=(DE)*(HL)
                                ;ALTERS A,B,C,D,E,H,L
                        IMULT:  PUSH    H               ;SAVE SECOND ARGUMENT IN CASE OF OVERFLOW
                                PUSH    D               ;SAVE FIRST ARGUMENT
                                CALL    IMULOV          ;FIX UP THE SIGNS
                                PUSH    B               ;SAVE THE SIGN OF THE RESULT
                                MOV     B,H             ;COPY SECOND ARGUMENT INTO (BC)
                                MOV     C,L
                                LXI     H,$CODE         ;ZERO (HL), THAT IS WHERE THE PRODUCT GOES
                                MVI     A,20            ;SET UP A COUNT
                        IMULT1: DAD     H               ;ROTATE PRODUCT LEFT ONE
                                JC      IMULT5          ;CHECK FOR OVERLFOW
```

```
 1793                                28500          XCHG                    ;ROTATE FIRST ARGUMENT LEFT ONE TO SEE IF
 1794                                28520          DAD     H               ; WE ADD IN (BC) OR NOT
 1795                                28540          XCHG
 1796                                28560          JNC     IMULT2          ;DON'T ADD IN ANYTHING
 1797                                28580          DAD     B               ;ADD IN (BC)
 1798                                28600          JC      IMULT5          ;CHECK FOR OVERFLOW
 1799                                28620  IMULT2: DCR     A               ;ARE WE DONE?
 1800                                28640          JNZ     IMULT1          ;NO, DO IT AGAIN
 1801                                28660          POP     B               ;WE ARE DONE, GET SIGN OF RESULT
 1802                                28680          POP     D               ;GET ORIGINAL FIRST ARGUMENT
 1803                                28700  IMLOIV: MOV     A,M             ;ENTRY FROM IDIV, IS RESULT .GE. 32768?
 1804                                28720          ORA     A
 1805                                28740          JM      IMULT3          ;IT IS, CHECK FOR SPECIAL CASE OF =32768
 1806                                28760          POP     D               ;RESULT IS OK, GET SECOND ARGUMENT OFF STACK
 1807                                28780          MOV     A,B             ;GET THE SIGN OF RESULT IN A
 1808                                28800          POP     B               ;NEGATE THE RESULT IF NECESSARY
 1809                                28820  IMULT3: XRI     200             ;IS RESULT 32768?
 1810                                28840          ORA     L               ;NOTE: IF WE GET HERE FROM IDIV, THE RESULT
 1811                                28860          JZ      IMULT4          ; MUST BE 32768, IT CANNOT BE GREATER
 1812                                28880          XCHG                    ;IT IS .GT. 32768, WE HAVE OVERFLOW
 1813                                28900          DW      1000,001        ;"LXI   B" OVER NEXT 2 BYTES
 1814                                28920  IMULT5: POP     B               ;GET SIGN OF RESULT OFF STACK
 1815                                28940          POP     H               ;GET THE ORIGINAL FIRST ARGUMENT
 1816                                28960          CALL    CONSIM          ;FLOAT IT
 1817                                28980          POP     H               ;GET THE ORIGINAL SECOND ARGUMENT
 1818                                29000          CALL    PUSHF           ;SAVE FLOATED FIRST ARUMENT
 1819                                29020          CALL    CONSIM          ;FLOAT SECOND ARGUMENT
 1820                                29040  FMULTI: POPR                    ;GET FIRST ARGUMENT OFF STACK, ENTRY FROM POLYX
 1821
 1822                                29060          JMP     FMULT           ;MULTIPLY THE ARGUMENTS USING SINGLE PRECISION
 1823                                29080  IMULT4: MOV     A,B             ;IS RESULT =32768 OR =32768?
 1824                                29100          ORA     A               ;GET ITS SIGN
 1825                                29120          POP     B               ;DISCARD ORIGINAL SECOND ARGUMENT
 1826                                29140          RM                      ;THE RESULT SHOULD BE NEGATIVE, IT IS OK
 1827                                29160          PUSH    D               ;IT IS POSITIVE, SAVE REMAINDER FOR MOD
 1828                                29180          CALL    CONSIM          ;FLOAT =32768
 1829                                29200          POP     D               ;GET MOD'S REMAINDER BACK
 1830                                29220          JMP     NEG             ;NEGATE =32768 TO GET 32768, WE ARE DONE
 1831                                29240
 1832                                29260
 1833                                29280          ;INTEGER DIVISION       (ML)=(DE)/(ML)
 1834                                29300          ;REMAINDER IS IN (DE), QUOTIENT IN (ML)
 1835                                29320          ;ALTERS A,B,C,D,E,H,L
 1836                                29340  IDIV:   MOV     A,H             ;CHECK FOR DIVISION BY ZERO
 1837                                29360          ORA     L
 1838                                29380          JZ      DV0ERR          ;WE HAVE DIVISION BY ZERO!!
 1839                                29400          CALL    IMULOV          ;FIX UP THE SIGNS
 1840                                29420          PUSH    B               ;SAVE THE SIGN OF THE RESULT
 1841                                29440          XCHG                    ;GET DENOMINATOR IN (HL)
 1842                                29460          CALL    INEGHL          ;NEGATE IT
 1843                                29480          MOV     B,H             ;SAVE NEGATED DENOMINATOR IN (BC)
 1844                                29500          MOV     C,L
 1845                                29520          LXI     H,$CODE         ;ZERO WHERE WE DO THE SUBTRACTION
```



```
 1846                                29540          MVI     A,21            ;SET UP A COUNT
 1847                                29560          PUSH    PSW             ;SAVE IT
 1848                                29580          ORA     A               ;CLEAR CARRY
 1849                                29600          JMP     IDIV3           ;GO DIVIDE
 1850                                29620  IDIV1:  PUSH    PSW             ;SAVE COUNT
 1851                                29640          PUSH    H               ;SAVE (ML) I.E. CURRENT NUMERATOR
 1852                                29660          DAD     B               ;SUBTRACT DENOMINATOR
 1853                                29680          JNC     IDIV2           ;WE SUBTRACTED TOO MUCH, GET OLD (ML) BACK
 1854                                29700          POP     PSW             ;THE SUBTRACTION WAS GOOD, DISCARD OLD (HL)
 1855                                29720          STC                     ;NEXT BIT IN QUOTIENT IS A ONE
 1856                                29740          DW      1000,076        ;"MVI   A" OVER NEXT BYTE
 1857                                29760  IDIV2:  POP     H               ;IGNORE THE SUBTRACTION, WE COULDN'T DO IT
 1858                                29780  IDIV3:  MOV     A,E             ;SHIFT IN THE NEXT QUOTIENT BIT
 1859                                29800          RAL
 1860                                29820          MOV     E,A
 1861                                29840          MOV     A,D             ;SHIFT THE MD
 1862                                29860          RAL
 1863                                29880          MOV     D,A
 1864                                29900          MOV     A,L             ;SHIFT IN THE NEXT BIT OF THE NUMERATOR
 1865                                29920          RAL
 1866                                29940          MOV     L,A
 1867                                29960          MOV     A,H             ;DO THE MD
 1868                                29980          RAL
 1869                                30000          MOV     H,A             ;SAVE THE MD
 1870                                30020          POP     PSW             ;GET COUNT BACK
 1871                                30040          DCR     A               ;ARE WE DONE?
 1872                                30060          JNZ     IDIV1           ;NO, DIVIDE AGAIN
 1873                                30080          XCHG                    ;GET QUOTIENT IN (ML), REMAINDER IN (DE)
 1874                                30100          POP     B               ;GET SIGN OF RESULT
 1875                                30120          PUSH    D               ;SAVE REMAINDER SO STACK WILL BE ALRIGHT
 1876                                30140          JMP     IMLDIV          ;CHECK FOR SPECIAL CASE OF 32768
 1877                                30160
 1878                                30180
 1879                                30200          ;GET READY TO MULTIPLY OR DIVIDE
 1880                                30220          ;ALTERS A,B,C,D,E,H,L
 1881                                30240  IMULOV: MOV     A,M             ;GET SIGN OF RESULT
 1882                                30260          XRA     D
 1883                                30280          MOV     B,A             ;SAVE IT IN B
 1884                                30300          CALL    INEGM           ;NEGATE SECOND ARGUMENT IF NECESARY
 1885                                30320          XCHG                    ;PUT (DE) IN (HL), FALL IN AND NEGATE FIRST
 1886                                30340                                  ; ARGUMENT IF NECESSARY
 1887                                30360
 1888                                30380
 1889                                30400          ;NEGATE H,L
 1890                                30420          ;ALTERS A,C,H,L
 1891                                30440  INEGM:  MOV     A,M             ;GET SIGN OF (ML)
 1892                                30460  INEGA:  ORA     A               ;SET CONDITION CODES
 1893                                30480          RP                      ;WE DON'T HAVE TO NEGATE, IT IS POSITIVE
 1894                                30500  INEGHL: XRA     A               ;CLEAR A
 1895                                30520          MOV     C,A             ;STORE A ZERO (WE USE THIS METHOD FOR ISUB)
 1896                                30540          SUB     L               ;NEGATE LO
 1897                                30560          MOV     L,A             ;SAVE IT
 1898                                30580          MOV     A,C             ;GET A ZERO BACK
```

```
                                30600           SBB     H               ;NEGATE MO
                                30620           MOV     H,A             ;SAVE IT
                                30640           RET                     ;ALL DONE
                                30660
                                30680
                                30700   ;INTEGER ABSOLUTE VALUE
                                30720   ;ALTERS A,B,C,D,E,H,L
                                30740   IABS:   LDA     FACLO+1         ;GET SIGN OF INTEGEN IN FAC
                                30760           ORA     A               ;CHECK ITS SIGN
                                30780           RP                      ;IT IS POSITIVE, LEAVE IT ALONE
                                30800                                   ;FALL INTO INEG AND NEGATE IT
                                30820
                                30840
                                30860   ;INTEGER NEGATION
                                30880   ;ALTERS A,B,C,D,E,H,L
                                30900   INEG:   LHLD    FACLO           ;GET THE INTEGER
                                30920           CALL    INEGHL          ;NEGATE IT
                                30940           SHLD    FACLO           ;STORE IT BACK IN THE FAC
                                30960           KRI     200             ;CHECK FOR SPECIAL CASE OF 32768
                                30980           ORA     L
                                31000           RNZ                     ;IT DID NOT OCCUR, EVERYTHING IS FINE
                                31020           KCMG                    ;WE HAVE IT, FLOAT 32768
                                31040           MVI     A,4             ;CHANGE VALTYP TO "SINGLE PRECISION"
                                31060           STA     VALTYP
                                31080   INEGAD: MVI     B,230           ;ENTRY FROM IADD, SET EXPONENT
                                31100           JMP     FLOATR          ;GO FLOAT THE NUMBER
                                31120
                                31140
                                31160   ;MOD OPERATOR
                                31180   ;(ML)1=(DE)=(UE)/(ML)*(ML),  (DE)=QUOTIENT
                                31200   ;ALTERS A,B,C,D,E,H,L
                                31220   MOD:    PUSH    D               ;SAVE (DE) FOR ITS SIGN
                                31240           CALL    IDIV            ;DIVIDE AND GET THE REMAINDER
                                31260           XCHG                    ;PUT REMAINDER IN (DE)
                                31280           MVI     A,2             ;SET VALTYP TO "INTEGER" IN CASE RESULT OF
                                31300           STA     VALTYP          ; THE DIVISION WAS 32768
                                31320           POP     PSW             ;GET THE SIGN OF (DE) BACK
                                31340           JMP     INEGA>          ;NEGATE THE REMAINDER IF NECESSARY
                                31360   PAGE
```



```
                                31380   SUBTTL  DOUBLE PRECISION ARITHMETIC ROUTINES
                                31400   IFE     LENGTH-2,4
                                31420   COMMENT %
                                31440           DOUBLE PRECISION ARITHMETIC CONVENTIONS
                                31460
                                31480   DOUBLE PRECISION NUMBERS ARE 8 BYTE QUANTITIES
                                31500   THE LAST 4 BYTES IN MEMORY ARE IN THE SAME FORMAT AS SINGLE PRECISION NUMBERS
                                31520   THE FIRST 4 BYTES ARE 32 MORE LOW ORDER BITS OF PRECISION
                                31540   THE LOWEST ORDER BYTE COMES FIRST IN MEMORY
                                31560
                                31580   CALLING CONVENTIONS:
                                31600   FOR ONE ARGUMENT FUNCTIONS:
                                31620           THE ARGUMENT IS IN THE FAC, THE RESULT IS LEFT IN THE FAC
                                31640   FOR TWO ARGUMENT OPERATIONS:
                                31660           THE FIRST ARGUMENT IS IN ARG=7,6,5,4,3,2,1,0  (NOTE: ARGLO=ARG=7)
                                31680           THE SECOND ARGUMENT IS IN THE FAC
                                31700           THE RESULT IS LEFT IN THE FAC
                                31720   VALTYP(DOUBLE PRECISION)=10 OCTAL
                                31740   %
                                31760
                                31780
                                31800           ;DOUBLE PRECISION SUBTRACTION   FAC:=ARG-FAC
                                3.820           ;ALTERS ALL REGISTERS
                                3.840   DSUB:   CALL    NEG             ;NEGATE THE SECOND ARGUMENT
                                3.860                                   ;FALL INTO DADD
                                3.880
                                31900
                                31920           ;DOUBLE PRECISION ADDITION      FAC:=ARG+FAC
                                31940           ;ALTERS ALL REGISTERS
                                31960   DADD:   LXI     H,ARG           ;GET  POINTER TO EXPONENT OF FIRST ARGUMENT
                                31980           MOV     A,M             ;CHECK IF IT IS ZERO
                                32000           ORA     A
                                32020           RZ                      ;IT IS, RESULT IS ALREADY IN FAC
                                32040           MOV     B,A             ;SAVE EXPONENT FOR UNPACKING
                                32060           DCX     H               ;POINT TO MO AND SIGN
                                32080           MOV     C,M             ;GET MO AND SIGN FOR UNPACKING
                                32100           LXI     D,FAC           ;GET POINTER TO EXPONENT OF SECOND ARGUMENT
                                32120           LDAX    D               ;GET EXPONENT
                                32140           ORA     A               ;SEE IF IT IS ZERO
                                32160           JZ      XMOVFA          ;IT IS, MOVE ARG TO FAC AND WE ARE DONE
                                32180           SUB     B               ;SUBTRACT EXPONENTS TO GET SHIFT COUNT
                                32200           JNC     DADD2           ;PUT THE SMALLER NUMBER IN FAC
                                32220           CMA                     ;NEGATE SHIFT COUNT
                                32240           INR     A
                                32260           PUSH    PSW             ;SAVE SHIFT COUNT
                                32280           PUSH    B               ;SAVE MO TO UNPACK LATER
                                32300           MVI     C,10            ;SWITCH FAC AND ARG, SET UP A COUNT
                                32320           INX     H               ;POINT TO ARG
                                32340   DADD11: LDAX    D               ;GET A BYTE OF THE FAC
                                32360           MOV     B,M             ;GET A BYTE OF ARG
                                32380           MOV     M,A             ;PUT THE FAC BYTE IN ARG
                                32400           MOV     A,B             ;PUT THE ARG BYTE IN A
                                32420           STAX    D               ;PUT THE ARG BYTE IN FAC
```

```
F4      MAC     23-AUG-64 06:08         DOUBLE PRECISION ARITHMETIC ROUTINES

 1991                           32448           DCX     D               ;POINT TO THE NEXT LO BYTE OF FAC
 1992                           32460           DCX     H               ;POINT TO THE NEXT LO BYTE OF ARG
 1993                           32480           DCR     C               ;ARE WE DONE?
 1994                           32500           JNZ     DADD1           ;NO, DO THE NEXT LO BYTE
 1995                           32520           POP     B               ;GET THE HO BACK
 1996                           32540           POP     PSW             ;GET THE SHIFT COUNT BACK
 1997                           32560   DADD2:  CPI     73              ;ARE WE WITHIN 56 BITS?
 1998                           32580           RNC                     ;NO, ALL DONE
 1999                           32600           PUSH    PSW             ;SAVE SHIFT COUNT
 2000                           32620           CALL    UNPACK          ;UNPACK THE NUMBERS
 2001                           32640           MOV     B,A             ;SAVE SUBTRACTION FLAG
 2002                           32660           MOV     A,C             ;SAVE THE UNPACKED HO
 2003                           32680           STA     ARG+1
 2004                           32700           POP     PSW             ;GET SHIFT COUNT
 2005                           32720           CALL    DSHFTR          ;SHIFT FAC RIGHT THE RIGHT NUMBER OF TIMES
 2006                           32740           ORA     B               ;GET SUBTRACTION FLAG, WERE ARG
 2007                           32760           JP      DADD3           ;SUBTRACT NUMBERS IF THEIR SIGNS ARE DIFFERENT
 2008                           32780           CALL    DADDAA          ;SIGNS ARE THE SAME, ADD THE NUMBERS
 2009                           32800           JNC     DROUND          ;ROUND THE RESULT IF NO CARRY
 2010                           32820           INR     M               ;WE HAVE OVERFLOW, ADD ONE TO THE EXPONENT
 2011                           32840           JZ      OVERR           ;CHECK FOR OVERFLOW
 2012                           32860           MVI     D,1             ;SHIFT NUMBER RIGHT ONE, SHIFT IN CARRY
 2013                           32880           CALL    DSHFRA
 2014                           32900           JMP     DROUND          ;ROUND THE RESULT
 2015                           32920   DADD3:  CMD     1000,076        ;**MVI   A*, SUBTRACT THE NUMBERS
 2016                           32940           SBB     M               ;GET THE SUBTRACT INSTRUCTION IN A
 2017                           32960           CALL    DADDA           ;SUBTRACT THE NUMBERS
 2018                           32980           INX     H               ;POINT TO THE UNPACKED SIGN
 2019                           33000           MOV     A,M             ;COMPLEMENT IT, SINCE THE FAC WAS SMALLER
 2020                           33020           CMA
 2021                           33040           MOV     M,A
 2022                           33060           CC      DNEGR           ;NEGATE THE RESULT IF IT WAS NEATIVE
 2023                           33080                                   ;FALL INTO DNORML
 2024                           33100
 2025                           33120
 2026                           33140           ;NORMALIZE FAC
 2027                           33160           ;ALTERS A,B,C,D,H,L
 2028                           33180   DNORML: XRA     A               ;CLEAR SHIFT COUNT
 2029                           33200   DNORM1: MOV     B,A             ;SAVE SHIFT COUNT
 2030                           33220           LDA     FAC+1           ;GET HO
 2031                           33240           ORA     A               ;SEE IF WE CAN SHIFT 8 LEFT
 2032                           33260           JNZ     DNORM5          ;WE CAN'T, SEE IF NUMBER IS NORMALIZED
 2033                           33280           LXI     H,OFACLO+1      ;WE CAN, GET POINTER TO LO
 2034                           33300           MVI     C,10            ;SET UP A COUNT
 2035                           33320   DNORM2: MOV     D,M             ;GET A BYTE OF FAC
 2036                           33340           MOV     M,A             ;PUT IN BYTE FROM LAST LOCATION, THE FIRST
                                                                        ; TIME THROUGH A IS ZERO
 2037                           33360           MOV     A,D             ;PUT THE CURRENT BYTE IN A FOR NEXT TIME
 2038                           33400           INX     H               ;INCREMENT POINTER TO NEXT HIGHER ORDER
 2039                           33420           DCR     C               ;ARE WE DONE?
 2040                           33440           JNZ     DNORM2          ;NO, DO THE NEXT BYTE
 2041                           33460           MOV     A,B             ;SUBTRACT 8 FROM SHIFT COUNT
 2042                           33480           SUI     10
```



```
F4      MAC     23-AUG-64 06:08         DOUBLE PRECISION ARITHMETIC ROUTINES

 2044                           33500           CPI     300             ;HAVE WE SHIFTED ALL BYTES TO ZERO?
 2045                           33520           JNZ     DNORM1          ;NO, TRY TO SHIFT 8 MORE
 2046                           33540           JMP     ZERO            ;YES, THE NUMBER IS ZERO
 2047                           33560   DNORM3: DCR     B               ;DECREMENT SHIFT COUNT
 2048                           33580           LXI     H,OFACLO+1      ;GET POINTER TO LO
 2049                           33600           CALL    DSHFLC          ;SHIFT THE FAC LEFT
 2050                           33620           ORA     A               ;SEE IF NUMBER IS NORMALIZED
 2051                           33640   DNORM5: JP      DNORM3          ;SHIFT FAC LEFT ONE IF IT IS NOT NORMALIZED
 2052                           33660           MOV     A,B             ;GET THE SHIFT COUNT
 2053                           33680           ORA     A               ;SEE IF NO SHIFTING WAS DONE
 2054                           33700           JZ      DROUND          ;NONE WAS, PROCEED TO ROUND THE NUMBER
 2055                           33720           LXI     H,FAC           ;GET POINTER TO EXPONENT
 2056                           33740           ADD     M               ;UPDATE IT
 2057                           33760           MOV     M,A             ;SAVE UPDATED EXPONENT
 2058                           33780           JNC     ZERO            ;UNDERFLOW, THE RESULT IS ZERO
 2059                           33800           RZ                      ;RESULT IS ALREADY ZERO, WE ARE DONE
 2060                           33820                                   ;FALL INTO DROUND AND ROUND THE RESULT
 2061                           33840
 2062                           33860
 2063                           33880           ;ROUND FAC
 2064                           33900           ;ALTERS A,B,H,L
 2065                           33920   DROUND: LDA     OFACLO+1        ;GET EXTRA BYTE TO SEE IF WE HAVE TO ROUND
 2066                           33940   DROUND: ORA     A               ;ENTRY FROM DDIV
 2067                           33960           CM      DROUNA          ;ROUND UP IF NECESSARY
 2068                           33980           LXI     H,FAC+1         ;GET POINTER TO UNPACKED SIGN
 2069                           34000           MOV     A,M             ;GET SIGN
 2070                           34020           ANI     200             ;ISOLATE SIGN BIT
 2071                           34040           DCX     H               ;POINT TO HO
 2072                           34060           DCX     H
 2073                           34080           XRA     M               ;PACK SIGN AND HO
 2074                           34100           MOV     M,A             ;PUT PACKED SIGN AND HO IN FAC
 2075                           34120           RET                     ;WE ARE DONE
 2076                           34140
 2077                           34160
 2078                           34180           ;SUBROUTINE FOR ROUND: ADD ONE TO FAC
 2079                           34200   DROUNA: LXI     H,OFACLO        ;GET POINTER TO LO, ENTRY FROM DINT
 2080                           34220           MVI     B,7             ;SET UP A COUNT
 2081                           34240   DRONA1: INR     M               ;INCREMENT A BYTE
 2082                           34260           RNZ                     ;RETURN IF THERE WAS NO CARRY
 2083                           34280           INX     H               ;INCREMENT POINTER TO NEXT HIGHER ORDER
 2084                           34300           DCR     B               ;HAVE WE INCREMENTED ALL BYTES
 2085                           34320           JNZ     DRONA1          ;NO, TRY THE NEXT ONE
 2086                           34340           INR     M               ;YES, INCREMENT THE EXPONENT
 2087                           34360           JZ      OVERR           ;CHECK FOR OVERFLOW
 2088                           34380           DCX     H               ;THE NUMBER OVERFLOWED ITS EXPONENT
 2089                           34400           MVI     M,200           ;PUT 200 IN HO
 2090                           34420           RET                     ;ALL DONE
 2091                           34440
 2092                           34460
 2093                           34480           ;ADD OR SUBTRACT 2 DBL QUANTITIES
 2094                           34500           ;ALTERS A,C,D,E,H,L
 2095                           34520   DADDD:  LXI     H,FBUFF+6+D17   ;ENTRY FROM DDIV
 2096                           34540           LXI     D,ARGLO         ;ADD OR SUBTRACT FBUFF+17 AND ARG
```

```
                                ;DO THE OPERATION
                                ;
                                ;**MVI   A*, ENTRY FROM DADD, DMULT
                                ;SETUP ADD INSTRUCTION FOR LOOP
                                ;GET POINTER TO ARG, ENTRY FROM DADD
                                ;GET POINTER TO FAC, ENTRY FROM FOUT
                                ;SET UP A COUNT
                                ;STORE THE ADD OR SUBTRACT INSTRUCTION
                                ;CLEAR CARRY
                                ;GET A BYTE FROM RESULT NUMBER
                                ;*THIS IS EITHER *ADC    M* OR *SBB      M*
                                ;SAVE THE CHANGED BYTE
                                ;INCREMENT POINTERS TO NEXT HIGHER ORDER BYTE
                                ;
                                ;ARE WE DONE?
                                ;NO, DO THE NEXT HIGHER ORDER BYTE
                                ;ALL DONE
                                ;
                                ;
                                ;NEGATE SIGNED NUMBER IN FAC
                                ;THIS IS USED BY DADD, DINT
                                ;ALTERS A,B,C,H,L
                                ;COMPLEMENT SIGN OF FAC
                                ;USE THE UNPACKED SIGN BYTE
                                ;SAVE THE NEW SIGN
                                ;GET POINTER TO LO
                                ;SET UP A COUNT
                                ;CLEAR CARRY AND GET A ZERO
                                ;SAVE ZERO IN C
                                ;GET A ZERO
                                ;NEGATE THE BYTE OF FAC
                                ;UPDATE FAC
                                ;INCREMENT POINTER TO NEXT HIGHER ORDER BYTE
                                ;ARE WE DONE?
                                ;NO, NEGATE THE NEXT BYTE
                                ;ALL DONE
                                ;
                                ;
                                ;SHIFT DBL FAC RIGHT ONE
                                ;A # SHIFT COUNT
                                ;ALTERS A,C,D,E,H,L
                                ;GET POINTER TO LO
                                ;PUT ZERO IN EXTRA LO ORDER BYTE
                                ;SEE IF WE CAN SHIFT 8 RIGHT
                                ;WE CAN*T, CHECK IF WE ARE DONE
                                ;ENTRY FROM DMULT, GET POINTER TO MQ
                                ;SHIFT A ZERO INTO THE MQ
                                ;SET UP A COUNT
                                ;SAVE A BYTE OF FAC
                                ;PUT THE LAST BYTE IN ITS PLACE
                                ;SET UP E FOR NEXT TIME THROUGH THE LOOP
                                ;POINT TO NEXT LOWER ORDER BYTE
                                ;ARE WE DONE?
```



```
                                ;NO, DO THE NEXT BYTE
                                ;YES, SEE IF WE CAN SHIFT OVER 8 MORE
                                ;CORRECT SHIFT COUNT
                                ;SAVE SHIFT COUNT IN D
                                ;CLEAR CARRY
                                ;ARE WE DONE?
                                ;YES
                                ;NO, GET POINTER TO LO, ENTRY FROM DADD, DMULT
                                ;SET UP A COUNT, ROTATE FAC ONE LEFT
                                ;GET A BYTE OF THE FAC
                                ;ROTATE IT LEFT
                                ;PUT THE UPDATED BYTE BACK
                                ;DECREMENT POINTER TO NEXT LOWER ORDER BYTE
                                ;ARE WE DONE?
                                ;NO, ROTATE THE NEXT LOWER ORDER BYTE
                                ;YES, SEE IF WE ARE DONE SHIFTING
                                ;
                                ;
                                ;ROTATE FAC LEFT ONE
                                ;ALTERS A,C,H,L
                                ;SET UP A COUNT
                                ;GET A BYTE OF FAC
                                ;ROTATE IT LEFT ONE
                                ;UPDATE BYTE IN FAC
                                ;INCREMENT POINTER TO NEXT HIGHER ORDER BYTE
                                ;ARE WE DONE?
                                ;NO, ROTATE THE NEXT BYTE
                                ;ALL DONE
                                ;
                                ;
                                ;DOUBLE PRECISION MULTIPLICATION        FAC1*ARG*FAC
                                ;ALTERS ALL REGISTERS
                                ;CHECK IF WE ARE MULTIPLYING BY ZERO
                                ;YES, ALL DONE, THE FAC IS ZERO
                                ;ADD EXPONENTS AND TAKE CARE OF SIGNS
                                ;ZERO FAC AND PUT FAC IN FBUFFR
                                ;PUT UNPACKED MD IN ARG
                                ;GET POINTER TO LO OF ARG
                                ;SET UP A COUNT
                                ;GET THE BYTE OF ARG TO MULTIPLY BY
                                ;INCREMENT POINTER TO NEXT HIGHER BYTE
                                ;CHECK IF WE ARE MULTIPLYING BY ZERO
                                ;SAVE POINTER TO ARG
                                ;WE ARE
                                ;SET UP A COUNT
                                ;SAVE COUNTERS
                                ;ROTATE MULTIPLIER RIGHT
                                ;SAVE IT
                                ;ADD IN OLD FAC IF BIT OF MULTIPIER WAS ONE
                                ;ROTATE PRODUCT RIGHT ONE
                                ;
                                ;GET MULTIPLIER IN A
                                ;GET COUNTERS BACK
```

```
                                36680           DCR     C               ;ARE WE DONE WITH THIS BYTE OF ARG?
                                36700           JNZ     DMULT3          ;NO, MULTIPLY BY THE NEXT BIT OF THE MULTIPLIER
                                36720
                                36740
                                36760
                                37000
                                37020
                                37040
                                37060
                                37080
                                37100
                                37120
                                37140
                                37160
                                37180
                                37200
                                37220
                                37240
                                37260
                                37280
                                37300
                                37320
                                37340
                                37360
                                37380
                                37400
                                37420
                                37440
                                37460
                                37480
                                37500
                                37520
                                37540
                                37560
                                37580
                                37600
                                37620
                                37640
                                37660
                                37680
                                37700
```

```
                                36780           POP     D               ;YES, GET POINTER INTO ARG BACK
                                36800           DCR     B               ;ARE WE DONE?
                                36820           JNZ     DMULT2          ;NO, MULTIPLY BY NEXT HIGHER ORDER BY OF ARG
                                36840           JMP     NORMAL          ;ALL DONE, NORMALIZE AND ROUND RESULT
                                36860           CALL    DSHFRM          ;SHIFT PRODUCT RIGHT ONE BYTE, WE ARE
                                36880           JMP     DMULT4          ; MULTIPLYIING BY ZERO
                                36900
                                36920
                                36940
                                36960
                                36980
                                37000
                                37020
                                37040
                                37060
                                37080
                                37100
                                37120
                                37140
                                37160
                                37180
                                37200
                                37220
                                37240
                                37260
                                37280
                                37300
                                37320
                                37340
                                37360
                                37380
                                37400
                                37420
                                37440
                                37460
                                37480
                                37500
                                37520
                                37540
                                37560
                                37580
                                37600
                                37620
                                37640
                                37660
                                37680
                                37700
```

```
                                36920           ;CONSTANT FOR DIV10, DDIV10
                                36940
                                36960
                                36980
                                37000
                                37020
                                37040
                                37060
                                37080
                                37100
                                37120
                                37140
                                37160
                                37180
                                37200
                                37220
                                37240
                                37260
                                37280
                                37300
                                37320
                                37340
                                37360
                                37380
                                37400
                                37420
                                37440
                                37460
                                37480
                                37500
                                37520
                                37540
                                37560
                                37580
                                37600
                                37620
                                37640
                                37660
                                37680
                                37700
```

```
DTEN:   000     ; 1000
        000
        000
        000
FTEN:   000     ; 10.0
        002
        042
        204
```

```
                                37000           ;DOUBLE PRECISION DIVIDE FAC BY 10
                                37020           ;ALTERS ALL REGISTERS
DDIV10: CALL    VMOVAF          ;SAVE THE FAC IN ARG
        LXI     H,DTEN          ;GET POINTER TO A DOUBLE PRECISION 10
        CALL    VMOVFM          ;MOVE TEN INTO THE FAC
                                ;FALL INTO DDIV AND DIVIDE BY TEN
                                37160
                                37180
                                37200
                                37220
                                37240
                                37260
                                37280
                                37300
                                37320
                                37340
                                37360
                                37380
                                37400
                                37420
                                37440
                                37460
                                37480
                                37500
                                37520
                                37540
                                37560
                                37580
                                37600
                                37620
                                37640
                                37660
                                37680
                                37700
```

```
                                37120           ;DOUBLE PRECISION DIVISION      FAC:=ARG/FAC
                                37140           ;ALTERS ALL REGISTERS
                                37160
                                37180
DDIV:   FSIGN                   ;CHECK FOR DIVISION BY ZERO
        JZ      DV0ERR          ;DON'T LET HIM DO IT
        CALL    MULDV8          ;SUBTRACT EXPONENTS AND CHECK SIGNS
        INR     M               ;ADD TWO TO EXPONENT TO CORRECT SCALING
        INR     M
        CALL    DMULDV          ;ZERO FAC AND PUT FAC IN FBUFFR
        LXI     H,ARG           ;GET POINTER TO THE EXTRA MO BYTE WE WILL USE
        MOV     M,C             ;ZERO IT
        MVI     B,0             ;ZERO FLAG TO SEE WHEN WE START DIVIDING
DDIV1:  XHD     1000,076        ;*MVI   A*, SUBTRACT FBUFFR FROM ARG
        SBB     M               ;GET SUBTRACT INSTRUCTION
        CALL    DADDD           ;DO THE SUBTRACTION
        LDAX    D               ;SUBTRACT FROM EXTRA MO BYTE
        SBB     C               ;HERE C=0
        CMC                     ;CARRY=1 IF SUBTRACTION WAS GOOD
        JC      DDIV2           ;WAS IT OK?
        XHD     1000,076        ;*MVI   A*  NO, ADD FBUFFR BACK IN
        ADC     M               ;GET ADD INSTRUCTION
        CALL    DADDD           ;DO THE ADDITION
        XRA     A               ;CLEAR CARRY
        XHD     1000,332        ;*JC* OVER NEXT TWO BYTES
DDIV2:  STAX    D               ;STORE THE NEW HIGHEST ORDER BYTE
```



```
                                37720           INR     B               ;INCREMENT FLAG TO SHOW WE COULD DIVIDE
                                37740           LDA     FAC+1           ;CHECK IF WE ARE DONE DIVIDING
                                37760           INR     A               ;SET SIGN FLAG WITHOUT AFFECTING CARRY
                                38000           DCR     A
                                38020           RAR                     ;PUT CARRY IN MSB FOR DROUND
                                38040           JM      DNORM8          ;WE ARE DONE, WE HAVE 57 BITS OF ACCURACY
                                38060           RAL                     ;GET OLD CARRY BACK WHERE IT BELONGS
                                38100           LXI     H,DFACLO        ;GET POINTER TO LO OF FAC
                                38120           MVI     C,7             ;SET UP A COUNT, SHIFT FAC LEFT ONE
                                38140           CALL    DSHFTL          ;SHIFT IN THE NEXT BIT IN THE QUOTIENT
                                38160           LXI     H,ARGLO         ;GET POINTER TO LO IN ARG
                                38200           CALL    DSHFLC          ;SHIFT DIVIDEND ONE LEFT
                                38220           MOV     A,B             ;IS THIS THE FIRST TIME AND WAS THE
                                38240           ORA     A               ; SUBTRACTION NOT GOOD? (B WILL GET
                                38260           JNZ     DDIV1           ; CHANGED ON THE FIRST OR SECOND SUBTRACTION)
                                38300           LXI     H,FAC           ;YES, SUBTRACT ONE FROM EXPONENT TO CORRECT
                                38320           DCR     M               ; SCALING
                                38340           JNZ     DDIV1           ;CONTINUE DIVIDING IF NO OVERFLOW
                                38360           JMP     OVERR           ;WE HAVE OVERFLOW!!
                                38400
                                38420
                                38440
                                38460
                                38480
                                38500           ;TRANSFER FAC TO FBUFFR FOR DMULT AND DDIV
                                38520           ;ALTERS A,B,C,D,E,H,L
DMULDV: MOV     A,C             ;PUT UNPACKED MO BACK IN ARG
        STA     ARG+1
        DCX     H               ;POINT TO MO OF FAC
        LXI     D,FBUFFR+0D23   ;POINT TO END OF FBUFFR
        MVI     B,7             ;SET UP A COUNT
        MVI     C,0             ;GET A ZERO TO FILL FAC WITH
DMLDV1: MOV     A,M             ;GET A BYTE FROM FAC
        STAX    D               ;PUT IT IN FBUFFR
        MOV     M,C             ;PUT A ZERO IN FAC
        DCX     D               ;POINT TO NEXT BYTE IN FBUFFR
        DCX     H               ;POINT TO NEXT LOWER ORDER BYTE IN FAC
        DCR     B               ;ARE WE DONE?
        JNZ     DMLDV1          ;NO, TRANSFER THE NEXT BYTE
        RET                     ;ALL DONE
```

```
                                38500           ;DOUBLE PRECISION MULTIPLY THE FAC BY 10
                                38520           ;ALTERS ALL REGISTERS
DMUL10: CALL    VMOVAF          ;SAVE THE FAC IN ARG
                                ;VMOVAF EXITS WITH (DE)=FAC+1
        XCHG                    ;GET THE POINTER INTO THE FAC IN (HL)
        DCX     H               ;POINT TO THE EXPONENT
        MOV     A,M             ;GET THE EXPONENT
        ADI     2               ;MULTIPLY FAC BY 4 BY ADDING 2 TO THE EXPONENT
        JC      OVERR           ;CHECK FOR OVERFLOW
        MOV     M,A             ;SAVE THE NEW EXPONENT
        PUSH    H               ;SAVE POINTER TO FAC
        CALL    DADD            ;ADD IN THE ORIGINAL FAC TO GET 5 TIMES  FAC
        POP     H               ;GET THE POINTER TO FAC BACK
        INR     M               ;ADD ONE TO EXPONENT TO GET 10 TIMES FAC
```

```
                                        38780           RNZ                     ;ALL DONE IF OVERFLOW DID NOT OCCUR
                                        38800           JMP     OVERR>          ;IT DID, GIVE THE APPROPRIATE MESSAGE
                                        38820   PAGE
```



```
                                        38840   $SBTTL  FLOATING POINT INPUT ROUTINE
                                        38860           ;ALTERS ALL REGISTERS
                                        38880           ;THE NUMBER IS LEFT IN FAC
                                        38900           ;AT ENTRY, (HL) POINTS TO THE FIRST CHARACTER IN A TEXT BUFFER.
                                        38920           ;THE FIRST CHARACTER IS ALSO IN A.  WE PACK THE DIGITS INTO THE FAC
                                        38940           ;AS AN INTEGER AND KEEP TRACK OF WHERE THE DECIMAL POINT IS,
                                        38960           ;C IS 377 IF WE HAVE NOT SEEN A DECIMAL POINT, 0 IF WE HAVE.
                                        38980           ;B IS THE NUMBER OF DIGITS AFTER THE DECIMAL POINT.
                                        39000           ;AT THE END, B AND THE EXPONENT (IN E) ARE USED TO DETERMINE HOW MANY
                                        39020           ;TIMES WE MULTIPLY OR DIVIDE BY TEN TO GET THE CORRECT NUMBER.
001531'                                 39040   FIN:
                                        39060           IFN     STRING,<
                                        39080           ;IF WE ARE CALLED BY VAL, THE SIGNS MAY NOT BE CRUNCHED
001531' 001000  000376                  39100           CPI     "-"             ;SEE IF NUMBER IS NEGATIVE
001532' 000000  000055
001533' 001000  000365                  39120           PUSH    PSW             ;SAVE SIGN
001536' 001000  000312                  39140           JZ      FIN1            ;IGNORE MINUS SIGN
001535' 000000  001545'
001536' 000000  001526'
001537' 001000  000376                  39160           CPI     "+"             ;IGNORE A LEADING SIGN
001540' 000000  000053
001541' 001000  000312                  39180           JZ      FIN1>
001542' 000000  001545'
001543' 000000  001535'
001546' 001000  000053                  39200           DCX     H               ;SET CHARACTER POINTER BACK ONE
001545'                                 39220   FIN1:
                                        39240           IFN     LENGTH-2,<
001545' 001000  000315                  39260           CALL    ZERO            ;CLEAR FAC
001546' 000000  000175'
001547' 000000  001542'
001550' 001000  000107                  39280           MOV     B,A             ;CLEAR FLAGS: B=DECIMAL PLACE COUNT
001551' 001000  000127                  39300           MOV     D,A             ;D=SIGN OF EXPONENT
001552' 001000  000137                  39320           MOV     E,A             ;E=EXPONENT
001553' 001000  000057                  39340           CMA
001554' 001000  000117                  39360           MOV     C,A             ;C="." FLAG
                                        39380           ;HERE TO GET THE NEXT DIGIT OF THE NUMBER, A DECIMAL POINT OR AN "E"
001555' 001000  000327                  39400   FINC:   CHRGET                  ;GET A CHARACTER
001556' 001000  000532                  39420           JC      FINDIG          ;DO WE HAVE A DIGIT?
001557' 000000  001702'
001560' 000-00  001546'
001561' 001000  000376                  39440           CPI     "."             ;TEST FOR DECIMAL POINT
001562' 000000  000056
001563' 001000  000312                  39460           JZ      FINDP
001564' 000000  001635'
001565' 000000  001357'
001566' 001000  000376                  39480           CPI     "E"             ;CHECK FOR BEGINNING OF EXPONENT
001567' 000-00  000105
001570' 001000  000302                  39500           JNZ     FINE            ;NONE OF THE ABOVE? SO END OF NUMBER
001571' 000000  001641'
001572' 000000  001564'
                                        39520           ;HERE TO CHECK FOR THE SIGN OF THE EXPONENT
001573' 001000  000327                  39540           CHRGET                  ;CHECK FOR ITS SIGN
                                        39560   IFN     STRING,<
```

```
2365  001574' 001000  000345          39580          PUSH    H               ;SAVE TEXT POINTER
2366  001575' 021020  000041          39600          LXI     H,FINEC         ;PUT FINEC ON STACK SO WE CAN JUMP
2367  001576' 000020  001621'
2368  001577' 000000  001571'
2369  001600' 001000  000303          39620          XTHL>                   ; TO IT IN LESS BYTES
2370  001601' 001000  000E25          39640          DCR     D               ;SET SIGN OF EXPONENT FLAG
2371  001602' 001000  000376          39660          CPI     MINUTK          ;NEGATIVE EXPONENT?
2372  001603' 000000  000002A
2373                                   39680  IFE     STRING,<
2374                                   39700          JZ      FINEC>
2375                                   39720  IFN     STRING,<
2376  001604' 001000  000510          39740          RZ
2377  001605' 001000  000376          39760          CPI     "-"
2378  001606' 000000  000055
2379  001607' 001000  000510          39780          RZ>
2380  001610' 001000  000E24          39800          INR     D               ;NO, RESET FLAG
2381                                   39820  IFN     STRING,<
2382  001611' 001000  000376          39840          CPI     "+"
2383  001612' 000000  000053
2384  001613' 001000  000310          39860          RZ>
2385  001614' 001000  000376          39880          CPI     PLUSTK          ;IGNORE "+"
2386  001615' 000000  000000*
2387                                   39900  IFE     STRING,<
2388                                   39920          JZ      FINEC>
2389                                   39940  IFN     STRING,<
2390  001616' 001000  000310          39960          RZ
2391  001617' 001000  000361          39980          POP     PSW>            ;GET FINEC OFF STACK
2392  001620' 001000  000E53          40000          DCX     H               ;CHECK IF LAST CHARACTER WAS A DIGIT
2393                                   40020          ;HERE TO GET THE NEXT DIGIT OF THE EXPONENT
2394  001621' 001000  000327          40040  FINEC:  CHRGET                  ;GET NEXT CHARACTER
2395  001622' 001000  000E3E          40060          JC      FINEDG          ;IS IT A DIGIT?
2396  001623' 000000  001746'
2397  001624' 000000  001576'
2398  001625' 001000  000E24          40080          INR     D               ;NO, EXPONENT ALL IN
2399  001626' 001000  000302          40100          JNZ     FINE            ;SET ITS SIGN
2400  001627' 000000  001641'
2401  001630' 000000  001623'
2402  001631' 001000  000E57          40120          XRA     A
2403  001632' 001000  000E23          40140          SUB     E
2404  001633' 001000  000137          40160          MOV     E,A
2405  001634' 001000  000014          40180          INR     C               ;MAKE SURE C IS NOT 377
2406                                   40200          ;HERE TO CHECK IF WE HAVE SEEN 2 DECIMAL POINTS AND SET THE DECIMAL
2407                                   40220          ; POINT FLAG
2408  001635' 001000  000014          40240  FINDP:  INR     C               ;DECIMAL POINTS!! == SET FLAG
2409  001636' 001000  000312          40260          JZ      FINC            ;CONTINUE SCANNING CHARACTERS
2410  001637' 000000  001555'
2411  001640' 000000  001627'
2412                                   40280                                  ;WE DON'T WANT TWO SO END OF NUMBER
2413                                   40300          ;HERE TO MULTIPLY OR DIVIDE BY 10 THE CORRECT NUMBER OF TIMES.
2414                                   40320          ;WE HAVE ALREADY READ IN ALL THE DIGITS.
2415  001641' 001000  000345          40340  FINE1:  PUSH    H               ;SAVE POINTER FOR LATER
2416  001642' 001000  0E3173          40360          MOV     A,E             ;EXPONENT=EXPONENT-# OF DECIMAL PLACES
2417  001643' 001000  000E20          40380          SUB     B
```



```
2418  001644' 001000  000364          40400  FINE2:  CP      FINMUL          ;MULTIPLY BY TEN IF EXPONENT IS POSITIVE
2419  001645' 000000  001672'
2420  001646' 000000  001637'
2421  001647' 001000  000E6C          40420          JP      FINE3           ;DIVIDE BY TEN IF EXPONENT IS NEGATIVE
2422  001650' 000000  001660'
2423  001651' 000000  001645'
2424  001652' 001000  000365          40440          PUSH    PSW             ;SAVE EXPONENT
2425  001653' 001000  000315          40460          CALL    DIV10           ;DIVIDE NUMBER BY TEN
2426  001654' 000000  000637'
2427  001655' 000000  001650'
2428  001656' 001000  000361          40480          POP     PSW             ;GET EXPONENT
2429  001657' 001000  000E74          40500          INR     A               ;INCREMENT IT
2430  001660' 001000  000302          40520  FINE3:  JNZ     FINE2           ;DO AGAIN IF WE ARE NOT DONE
2431  001661' 000000  001644'
2432  001662' 000000  001654'
2433                                   40540  IFE     STRING,<
2434                                   40560          POP     H>              ;GET CHARACTER POINTER
2435                                   40580  IFN     STRING,<
2436  001663' 001000  000321          40600          POP     D               ;GET CHARACTER POINTER
2437  001664' 001000  000361          40620          POP     PSW             ;GET SIGN
2438  001665' 001000  000314          40640          CZ      NEG             ;NEGATE IF NECESSARY
2439  001666' 000000  001175'
2440  001667' 000000  001661'
2441  001670' 001000  000353          40660          XCHG>                   ;GET CHARACTER POINTER IN (HL)
2442  001671' 001000  000311          40680          RET>                    ;ALL DONE
2443                                   40700  IFE     LENGTH-2,<
2444                                   40720          XCHG                    ;SAVE THE TEXT POINTER IN (DE)
2445                                   40740          LXI     B,377*$CODE     ;CLEAR FLAGS: B=DECIMAL PLACE COUNT
2446                                   40760                                  ;C="-" FLAG
2447                                   40780          MOV     H,C             ;ZERO (HL)
2448                                   40800          MOV     L,C
2449                                   40820          CALL    CONISS          ;ZERO FAC, SET VALTYP TO "INTEGER"
2450                                   40840          XCHG                    ;GET THE TEXT POINTER BACK IN (HL)
2451                                   40860          ;HERE TO CHECK FOR A DIGIT, A DECIMAL POINT, "E" OR "D"
2452                                   40880  FINC:   CHRGET                  ;GET THE NEXT CHARACTER OF THE NUMBER
2453                                   40900          JC      FINDIG          ;WE HAVE A DIGIT
2454                                   40920          CPI     "."             ;CHECK FOR A DECIMAL POINT
2455                                   40940          JZ      FINDP           ;WE HAVE ONE, I GUESS
2456                                   40960          CPI     "E"             ;CHECK FOR A SINGLE PRECISION EXPONENT
2457                                   40980          JZ      FINEX           ;WE HAVE A SINGLE PRECISION NUMBER
2458                                   41000          CPI     "D"             ;CHECK FOR A DOUBLE PRECISION EXPONENT
2459                                   41020          JNZ     FINE            ;WE DON'T HAVE ONE, THE NUMBER IS FINISHED
2460                                   41040          ORA     A               ;DOUBLE PRECISION NUMBER == TURN OFF ZERO FLAG
2461                                   41060  FINEX:  CALL    FINFRC          ;FORCE THE FAC TO BE SNG OR DBL
2462                                   41080          PUSH    H               ;SAVE THE TEXT POINTER
2463                                   41100          LXI     H,FINEC         ;GET ADDRESS TO JUMP TO, THIS IS TO SAVE BYTES
2464                                   41120          XTHL                    ;PUT IT ON STACK AND GET TEXT POINTER
2465                                   41140          ;HERE TO CHECK FOR THE SIGN OF THE EXPONENT
2466                                   41160          CHRGET                  ;GET THE FIRST CHARACTER OF THE EXPONENT
2467                                   41180          DCR     D               ;SET SIGN OF EXPONENT TO MINUS
2468                                   41200          CPI     MINUTK          ;CHECK IF THE EXPONENT IS NEGATIVE
2469                                   41220          RZ                      ;IT IS
2470                                   41240          CPI     "-"             ;THIS IS IN CASE WE ARE CALLED BY VAL
```

```
                                        RZ
                                        INR     D               ;EXPONENT IS STILL POSITIVE, RESET FLAG
                                        CPI     PLUSTK          ;IGNORE A LEADING PLUS SIGN
                                        RZ
                                        CPI     '-'
                                        RZ
                                        DCX     H               ;THE FIRST CHARACTER WAS NOT A SIGN, GO BACK
                                        ; AND CHECK FOR A DIGIT
                                        POP     PSW             ;POP FINEC OFF THE STACK, WE NO LONGER NEED IT
                                        ;HERE TO GET THE NEXT DIGIT OF THE EXPONENT
                        FINEC:  CHRGET                          ;GET THE NEXT CHARATER
                                JC      FINEDG                  ;PACK THE NEXT DIGIT INTO THE EXPONENT
                                INR     D                       ;IT WAS NOT A DIGIT, PUT THE CORRECT SIGN ON
                                JNZ     FINE                    ; THE EXPONENT, IT IS POSITIVE
                                XRA     A                       ;THE EXPONENT IS NEGATIVE
                                SUB     E                       ;NEGATE IT
                                MOV     E,A                     ;SAVE IT AGAIN
                                ;HERE TO FINISH UP THE NUMBER
                        FINE:   LDA     VALTYP                  ;FINISH UP -- WHAT KIND OF A NUMBER IS IT?
                                CPI     2
                                JNZ     FINEF                   ;IT IS A FLOATING POINT ONE
                                ;HERE TO FINISH UP AN INTEGER
                                POP     PSW                     ;IT IS AN INTEGER, GET ITS SIGN
                                XCHG                            ;SAVE THE TEXT POINTER IN (DE)
                                CZ      INEG                    ;NEGATE IT IF NECESSARY
                                XCHG                            ;GET THE TEXT POINTER BACK IN (HL)
                                RET                             ;ALL DONE
                                ;HERE TO FINISH UP A FLOATING POINT NUMBER
                        FINEF:  PUSH    H                       ;SAVE THE TEXT POINTER
                                MOV     A,E                     ;FIND OUT HOW MANY TIMES WE HAVE TO MULTIPLY
                                SUB     B                       ; OR DIVIDE BY TEN
                                ;HERE TO MULTIPLY OR DIVIDE BY TEN THE CORRECT NUMBER OF TIMES
                        FINEF2: CP      FINMUL                  ;MULTIPLY IF WE HAVE TO
                                CM      FINDIV                  ;DIVIDE IF WE HAVE TO
                                JNZ     FINEF2                  ;MULTIPLY OR DIVIDE AGAIN IF WE ARE NOT DONE
                                ;HERE TO PUT THE CORRECT SIGN ON THE NUMBER
                                POP     D                       ;GET THE TEXT POINTER
                                POP     PSW                     ;GET THE SIGN
                                CZ      NEG                     ;NEGATE IF NECESSARY
                                XCHG                            ;GET THE TEXT POINTER IN (HL)
                                LDA     VALTYP                  ;WE WANT -32768 TO BE AN INT, BUT UNTIL NOW
                                CPI     4                       ; IT WOULD BE A SNG
                                RNZ                             ;IT IS NOT SNG, SO IT IS NOT -32768
                                PUSH    H                       ;WE HAVE A SNG, SAVE TEXT POINTER
                                LXI     H,POPHRT                ;GET ADDRESS THAT POPS H OFF STACK BECAUSE
                                PUSH    H                       ; CONI32 DOES FUNNY THINGS WITH THE STACK
                                CALL    CONI32                  ;CHECK IF WE HAVE -32768
                                RET                             ;WE DON'T, POPHRT IS STILL ON THE STACK SO
                                                                ; WE CAN JUST RETURN


                                ;HERE TO CHECK IF WE HAVE SEEN 2 DECIMAL POINTS AND SET THE DECIMAL
                                ; POINT FLAG
                        FINDP:  INR     C                       ;SET THE FLAG
```



```
                                JNZ     FINEF                   ;WE HAD 2 DECIMAL POINTS, NOW WE ARE DONE
                                CALL    FINFRC                  ;THIS IS THE FIRST ONE, CONVERT FAC TO SNG
                                JMP     FINC                    ;CONTINUE LOOKING FOR DIGITS


                                ;FORCE THE FAC TO BE SNG OR DBL
                                ;IF THE ZERO FLAG IS ON, THEN FORCE THE FAC TO BE SNG
                                ;IF THE ZERO FLAG IS OFF, FORCE THE FAC TO BE DBL
                        FINFRC: PUSH    H                       ;SAVE TEXT POINTER
                                PUSH    D                       ;SAVE EXPONENT INFORMATION
                                PUSH    B                       ;SAVE DECIMAL POINT INFORMATION
                                PUSH    PSW                     ;SAVE WHAT WE WANT THE FAC TO BE
                                CZ      FRCSNG                  ;CONVERT TO SNG IF WE HAVE TO
                                POP     PSW                     ;GET TYPE FLAG BACK
                                CNZ     FRCDBL                  ;CONVERT TO DBL IF WE HAVE TO
                                POP     B                       ;GET DECIMAL POINT INFORMATION BACK
                                POP     D                       ;GET EXPONENT INFORMATION BACK
                                POP     H                       ;GET TEXT POINTER BACK
                                RET>                            ;ALL DONE


                                ;THIS SUBROUTINE MULIPLIES BY TEN ONCE,
                                ;IT IS A SUBROUTINE BECAUSE IT SAVES BYTES WHEN WE CHECK IF A IS ZERO
                        FINMUL: RZ                              ;RETURN IF EXPONENT IS ZERO, ENTRY FROM FOUT
                        FINMLT: PUSH    PSW                     ;SAVE EXPONENT, ENTRY FROM FOUT
                                IFN     LENGTH=2,<
                                CALL    MUL10>                  ;MULTIPLY BY TEN

                                IFE     LENGTH=2,<
                                LDA     VALTYP                  ;SEE WHAT KIND OF NUMBER WE HAVE
                                CPI     4
                                PUSH    PSW                     ;SAVE THE TYPE
                                CZ      MUL10                   ;WE HAVE A SNG, MULTIPLY BY 10.0
                                POP     PSW                     ;GET THE TYPE BACK
                                CNZ     DMUL10>                 ;WE HAVE A DBL, MULTIPLY BY 1000
                                POP     PSW                     ;GET EXPONENT
                        DCRART: DCR     A                       ;DECREASE IT
                                RET                             ;ALL DONE

                                IFE     LENGTH=2,<
                        FINDIV: PUSH    PSW                     ;WE HAVE TO DIVIDE -- SAVE COUNT
                                LDA     VALTYP                  ;SEE WHAT KIND OF NUMBER WE HAVE
                                CPI     4
                                PUSH    PSW                     ;SAVE THE TYPE
                                CZ      DIV10                   ;WE HAVE A SNG NUMBER
                                POP     PSW                     ;GET THE TYPE BACK
                                CNZ     DDIV10                  ;WE HAVE A DBL NUMBER
                                POP     PSW                     ;GET COUNT BACK
                                INR     A                       ;UPDATE IT
                                RET>


                                ;HERE TO PACK THE NEXT DIGIT OF THE NUMBER INTO THE FAC
                                ;WE MULTIPLY THE FAC BY TEN AND ADD IN THE NEXT DIGIT
001702'                 FINDIG:
```

```
                                43340   IFN     LENGTH=2,<
                                43360           PUSH    D               ;DIGITS: SAVE EXPONENT INFORMATION
                                43380           MOV     D,A             ;PROTECT DIGIT FROM BELOW
                                43400           MOV     A,B             ;INCREMENT DECIMAL PLACE COUNT
                                43420           ADC     C               ; IF PAST THE DECIMAL POINT
                                43440           MOV     B,A
                                43460           PUSH    B               ;SAVE NECESSARY DATA
                                43480           PUSH    H
                                43500           PUSH    D               ;SAVE DIGIT
                                43520           CALL    MUL10           ;MULTIPLY OLD NUMBER BY 10
                                43540           POP     PSW             ;GET NEXT DIGIT
                                43560           SUI     "0"             ;SUBTRACT OFF ASCII CODE
                                43580   IFE     EXTFNC,<
                                43600           CALL    PUSHF           ;PUT NUMBER ON STACK
                                43620           CALL    FLOAT           ;CONVERT TO FLOATING POINT NUMBER
                                43640           POPR
                                43660           CALL    FADD>           ;ADD IN NEXT DIGIT
                                43680   IFN     EXTFNC,<
                                43700           CALL    FINLOG>

                                43720           POP     H               ;RECALL DATA
                                43740           POP     B
                                43760           POP     D
                                43780           JMP     FINC>           ;GET NEXT CHARACTER

                                43800   IFE     LENGTH=2,<
                                43820           PUSH    D               ;SAVE EXPONENT INFORMATION
                                43840           MOV     A,B             ;INCREMENT DECIMAL PLACE COUNT IF WE ARE
                                43860           ADC     C               ; PAST THE DECIMAL POINT
                                43880           MOV     B,A
                                43900           PUSH    B               ;SAVE DECIMAL POINT INFORMATION
                                43920           PUSH    H               ;SAVE TEXT POINTER
                                43940           MOV     A,M             ;GET THE DIGIT
                                43960           SUI     "0"             ;CONVERT IT TO ASCII
                                43980           PUSH    PSW             ;SAVE THE DIGIT
                                44000           LDA     VALTYP          ;SEE WHAT KIND OF A NUMBER WE HAVE
                                44020           CPI     4
                                44040           JNC     FINDBV          ;WE DO NOT HAVE AN INTEGER
                                44060           ;HERE TO PACK THE NEXT DIGIT OF AN INTEGER
                                44080           LHLD    FACLO           ;WE HAVE AN INTEGER, GET IT IN (HL)
                                44100           LXI     D,-(32767+1)    ;SEE IF WE WILL OVERFLOW
                                44120           COMPAR                  ;COMPAR RETURNS WITH CARRY ON IF
                                44140           JNC     FINDB2          ; (HL) .LT. (DE), SO THE NUMBER IS TOO BIG
                                44160           MOV     D,H             ;COPY (HL) INTO (DE)
                                44180           MOV     E,L
                                44200           DAD     H               ;MULTIPLY (HL) BY 2
                                44220           DAD     H               ;MULTIPLY (HL) BY 2, (HL) NOW IS 4*(DE)
                                44240           DAD     D               ;ADD IN OLD (HL) TO GET 5*(DE)
```



```
                                44260           DAD     H               ;MULTIPLY BY 2 TO GET TEN TIMES THE OLD (HL)
                                44280           POP     PSW             ;GET THE DIGIT
                                44300           MOV     C,A             ;SAVE IT SO WE CAN USE DAD, B IS ALREADY ZERO
                                44320           DAD     B               ;ADD IN THE NEXT DIGIT
                                44340           MOV     A,H             ;CHECK FOR OVERFLOW
                                44360           ORA     A               ;OVERFLOW OCCURED IF THE MSB IS ON
                                44380           JM      FINDG1          ;WE HAVE OVERFLOW!!
                                44400           SHLD    FACLO           ;EVERYTHING IS FINE, STORE THE NEW NUMBER
                                44420   FINDGE: POP     H               ;ALL DONE, GET TEXT POINTER BACK
                                44440           POP     B               ;GET DECIMAL POINT INFORMATION BACK
                                44460           POP     D               ;GET EXPONENT INFORMATION BACK
                                44480           JMP     FINC            ;GET THE NEXT CHARACTER
                                44500           ;HERE TO HANDLE 32768, 32769
                                44520   FINDG1: MOV     A,C             ;GET THE DIGIT
                                44540           PUSH    PSW             ;PUT IT BACK ON THE STACK
                                44560           ;HERE TO CONVERT THE INTEGER DIGITS TO SINGLE PRECISION DIGITS
                                44580   FINDG2: CALL    CONSI           ;CONVERT THE INTEGER TO SINGLE PRECISION
                                44600           XRA     A               ;DO NOT TAKE THE FOLLOWING JUMP
                                44620           ;HERE TO DECIDE IF WE HAVE A SINGLE OR DOUBLE PRECISION NUMBER
                                44640   FINDGV: JNZ     FIND6D          ;FALL THROUGH IF VALTYP WAS 4 I.E, SNG PREC
                                44660           MVIXI   224,168,044,000 ;GET 100000?, DO WE HAVE 7 DIGITS ALREADY?
                                44680           CALL    FCOMP           ;IF SO, FAC .GE. 1000000
                                44700           JP      FINDG3          ;WE DO, CONVERT TO DOUBLE PRECISION
                                44720           POP     PSW             ;GET THE NEXT DIGIT
                                44740           CALL    FINLOG          ;PACK IT INTO THE FAC
                                44760           JMP     FINDGE          ;GET FLAGS OFF STACK AND WE ARE DONE
                                44780           ;HERE TO CONVERT A 7 DIGIT SINGLE PRECISION NUMBER TO DOUBLE PRECISION
                                44800   FINDG3: CALL    CONDS           ;CONVERT SINGLE TO DOUBLE PRECISION
                                44820           ;HERE TO PACK IN THE NEXT DIGIT OF A DOUBLE PRECISION NUMBER
                                44840   FINDG0: CALL    DMUL10          ;MULTIPLY THE FAC BY 10
                                44860           CALL    VMOVAF          ;SAVE THE FAC IN ARG
                                44880           POP     PSW             ;GET THE NEXT DIGIT
                                44900           CALL    FLOAT           ;CONVERT THE DIGIT TO SINGLE PRECISION
                                44920           CALL    CONDS           ;NOW, CONVERT THE DIGIT TO DOUBLE PRECISION
                                44940           CALL    DADD            ;ADD IN THE DIGIT
                                44960           JMP     FINDGE>         ;GET THE FLAGS OFF THE STACK AND WE ARE DONE

                                45000   IFN     EXTFNC,<
                                45020           ;SUBROUTINE FOR FIN, LOG
                                45040   FINLOG: CALL    PUSHF           ;SAVE FAC ON STACK

                                45060           CALL    FLOAT           ;CONVERT A TO A FLOATING POINT NUMBER

                                45080   IFN     LENGTH=2,<
                                45100           JMP     FADDT>          ;ADD IT IN

                                45120   IFE     LENGTH=2,<
                                45140           POPR                    ;GET PREVIOUS NUMBER OFF STACK
                                45160           JMP     FADD>>          ;ADD IT IN
```

```
F4      MAC     23-AUG-64 06108         FLOATING POINT INPUT ROUTINE

                                        ;HERE WE PACK IN THE NEXT DIGIT OF THE EXPONENT
                                        ;WE MUTIPLY THE OLD EXPONENT BY TEN AND ADD IN THE NEXT DIGIT
                                        ;NOTE: EXPONENT OVERFLOW IS NOT CHECKED FOR
001742' 001000  000173          FINEDG: MOV     A,E             ;EXPONENT DIGIT -- MULTIPLY EXPONENT BY 10
001743' 001000  000027          RLC                             ;FIRST BY 4
001744' 001000  000027          RLC
                                ADD     E               ;ADD 1 TO MAKE 5
001745' 001000  000023          RLC                             ;NOW DOUBLE TO GET 10
001746' 001000  000027          ADD     M               ;ADD IT IN
001747' 001000  000206          SUI     "0"             ;SUBTRACT OFF ASCII CODE
001750' 001000  000326          MOV     E,A             ;STORE EXPONENT
001751' 000000  000060          JMP     FINEC           ;CONTINUE
001752' 001000  000137
001753' 001000  000303
001754' 000000  001621'
001755' 000000  001746'
                        PAGE
```



```
F4      MAC     23-AUG-64 06108         FLOATING POINT OUTPUT ROUTINE

                        SUBTTL  FLOATING POINT OUTPUT ROUTINE
                                ;ENTRY TO LINPRT
001756' 001000  000345  INPRT:  PUSH    H               ;SAVE LINE NUMBER
001757' 001000  000041          LXI     H,INTXT##       ;PRINT MESSAGE
001760' 000000  000000#
001761' 000000  001754'
001762' 001000  000315          CALL    STROUT
001763' 000000  000000#
001764' 000000  001760'
001765' 001000  000341          POP     H               ;FALL INTO LINPRT

                                ;PRINT THE 2 BYTE NUMBER IN H,L
                                ;ALTERS ALL REGISTERS
001766'                 LINPRT: IFN     LENGTH-2,<
001766' 001000  000353          XCHG                    ;SET UP REGISTERS FOR FLOATH
001767' 001000  000257          XRA     A
001770' 001000  000006          MVI     B,230
001771' 000000  000230
001772' 001000  000315          CALL    FLOATR#         ;CONVERT TO FLOATING POINT
001773' 000000  001155'
001774' 000000  001763'

                        IFE     LENGTH-2,<
                                CALL    CONISS          ;PUT THE LINE NUMBER IN THE FAC AS AN INTEGER
                                XRA     A               ;SET FORMAT TO FREE FORMAT
                                CALL    FOUINI#         ;SET UP THE SIGN
001775' 001000  000041          LXI     H,STRUO1##      ;PUT PRINT STRING ADDRESS ON STACK SO WE WILL
001776' 000000  000000#
001777' 000000  001773'
002000' 001000  000345          PUSH    H               ; RETURN TO IT AND DO AN "INX   M"
                                                        ;THIS GETS RID OF THE SPACE FOR THE SIGN AT
                                                        ; THE BEGINNING OF A LINE NUMBER
                                                        ;FALL INTO FOUT
                                IFE     LENGTH-2,<
                                PUSH    B               ;PUT DUMMY FIELD LENGTHS ON STACK -- (CALL FOUT2
                                JMP     STRAUI##        ;PRINT THE NUMBER

                                ;FLOATING OUTPUT OF FAC
                                ;ALTERS ALL REGISTERS
                                ;THE ORIGINAL CONTENTS OF THE FAC IS LOST
                        IFN     LENGTH-2,<
002001' 001000  000041  FOUT:   LXI     H,FBUFFR        ;GET BEGINING OF CHARACTER BUFFER
002002' 000000  000000#
002003' 000000  001776'
002004' 001000  000345          PUSH    H               ;SAVE IT FOR WHEN WE RETURN
                                ;PUT THE SIGN OF THE NUMBER IN THE BUFFER AND MAKE IT POSITIVE
002005' 001000  000357          FSIGN                   ;GET SIGN OF NUMBER
002006' 001000  000066          MVI     M," "           ;PRINT SPACE IF POSITIVE
002007' 000000  000040
002010' 001000  000362          JP      FOUT1
002011' 000000  002615'
```

```
 2752  002214' 007200  002200'
 2753  002215' 001000  000066          46260           MVI     M,"-"           ;PRINT A MINUS SIGN IF NEGATIVE
 2754  002216' 000000  000055
 2755  002217' 001000  000060          46280   FOUT11  INX     H               ;INCREMENT POINTER TO NEXT CHRACTER POSITION
 2756  002220' 001000  000066          46300           MVI     M,"0"           ;PUT A ZERO IN BUFFER IN CASE NUMBER=0
 2757  002221' 000000  000060
 2758  002222' 001000  000314          46320           JZ      FOUT19          ;DO IT IF THE NUMBER IS ZERO
 2759  002223' 000000  002266'
 2760  002224' 000000  002411'
 2761  002225' 001000  000345          46340           PUSH    H               ;SAVE BUFFER POINTER
 2762  002226' 001000  000374          46360           CM      NEG             ;NEGATE NUMBER IF NEGATIVE
 2763  002227' 000000  001175'
 2764  002230' 000000  002021'
 2765
 2766                                   46400   ;HERE WE GET THE FAC IN THE RANGE 100000 .LE. FAC .LE. 999999 AND ROUND IT TO
 2767                                   46420   ;AN INTEGER.  WE KEEP A COUNT OF HOW MANY TIMES WE MULTIPLY OR DIVIDE BY TEN
 2768                                   46440   ;SO WE KNOW WHAT THE EXPONENT WILL BE.  THE FAC IS THEN CONVERTED TO AN
 2769                                   46460   ;INTEGER IN C,D,E.  WE USE A TABLE OF POWERS OF TEN TO CALCULATE EACH DIGIT.
 2770                                   46480   ;THIS ALGORITHM IS USED FOR SPEED.
 2771  002231' 001000  000257          46500           XRA     A               ;PUT TEN'S EXPONENT COUNT ON STACK
 2772  002232' 001000  000365          46520           PUSH    PSW
 2773  002233' 001000  000315          46540           CALL    FOUTC8          ;SEE IF NUMBER IS TOO BIG OR TOO SMALL
 2774  002234' 000000  002274'
 2775  002235' 000000  002225'
 2776  002236' 001000  000001          46560   FOUT31  LXI     B21,103,117,370 ;IS NUMBER .LE. 99999.9499?  IT IS TOO SMALL
 2777  002237' 000000  000103
 2778  002240' 000000  000021
 2779  002241' 001000  000021          46580           LXI     D,241,200,377
 2780  002242' 000000  000370
 2781  002243' 000000  000377
 2782  002244' 001000  000315          46600           CALL    FCOMP           ;FCOMP RETURNS 377, 0 OR 1 IN A, SO THE
 2783  002245' 000000  001311'
 2784  002246' 000000  002032'
 2785                                   46620                                   ; PARITY WILL BE ODD IFF 1 IS RETURNED
 2786  002247' 001000  000342          46620           JPO     FOUT5           ;NO, NUMBER IS IN RANGE
 2787  002250' 000000  002271'
 2788  002251' 000000  002043'
 2789  002252' 001000  000361          46640           POP     PSW
 2790  002253' 001000  000315          46660           CALL    FINMLT          ;YES, MULTIPLY IT BY TEN TO GET
 2791  002254' 000000  001673'
 2792  002255' 000000  002246'
 2793  002256' 001000  000365          46680           PUSH    PSW             ; IT IN RANGE
 2794  002257' 001000  000303          46700           JMP     FOUT3           ;SEE IF NUMBER IS NOW IN RANGE
 2795  002260' 000000  002034'
 2796  002261' 000000  002052'
 2797  002262' 001000  000315          46720   FOUT9:  CALL    DIV10           ;NO, DIVIDE NUMBER BY TEN, IT IS TOO BIG
 2798  002263' 000000  000637'
 2799  002264' 000000  002256'
 2800  002265' 001000  000361          46740           POP     PSW             ;ADD ONE TO EXPONENT
 2801  002266' 001000  000074          46760           INR     A
 2802  002267' 001000  000365          46780           PUSH    PSW
 2803  002270' 001000  000315          46800           CALL    FOUTC8          ;IS NUMBER .LE. 999999.499?
 2804  002271' 000000  002274'
```



```
 2805  002272' 000000  002261'
 2806                                   46820                                   ;YES, NUMBER IS IN PRINTING RANGE, I.E.
 2807                                   46840                                   ; ALL DIGITS TO BE PRINTED ARE THE INTEGER PART
 2808
 2809  002274' 001000  000315          46860   FOUT5:  CALL    FADDM           ;ROUND NUMBER TO NEAREST INTEGER
 2810  002277' 000000  000006'
 2811  002300' 000000  002267'
 2812  002301' 001000  000074          46880           INR     A               ;MAKE A NON-ZERO, SINCE NUMBER IS POSITIVE
 2813                                   46900                                   ; AND NON-ZERO, ROUND WILL EXIT WITH THE MO
 2814                                   46920                                   ; IN A, SO THE MSB WILL ALWAYS BE ZERO AND
 2815                                   46940                                   ; ADDING ONE WILL NEVER CAUSE A TO BE ZERO
 2816  002302' 001000  000315          46960           CALL    GINT            ;GET INTEGER PART IN C,D,E
 2817  002305' 000000  001372'
 2818  002306' 000000  002267'
 2819  002307' 001000  000315          46980           CALL    MOVFR           ;SAVE NUMBER IN FAC
 2820  002312' 000000  001425'
 2821  002313' 000000  002278'
 2822                                   47000                                   ;DECIDE IF THE NUMBER SHOULD BE PRINTED IN FIXED OR FLOATING NOTATION
 2823  002316' 001000  000001          47020           LXI     B,2*400+6*$CODE ;SET DECIMAL POINT COUNT FOR E NOTATION
 2824  002317' 000000  001406*
 2825  002320' 000000  002101'
 2826                                   47040                                   ;C = DIGIT COUNT
 2827  002321' 001000  000361          47060           POP     PSW             ;GET EXPONENT
 2828                                   47080           ADD     C               ;[redacted] NUMBER [redacted]BE PRINTED IN E NOTATION?
 2829  002322' 001000  000201                                                  ;SINCE E+N
 2830  002323' 001000  000374          47100           JM      FOUT6           ;YES, IT IS .LT. .1
 2831  002324' 000000  002124'
 2832  002325' 000000  002104'
 2833  002326' 001000  000376          47120           CPI     7
 2834  002327' 000000  000007
 2835  002330' 001000  000322          47140           JNC     FOUT6           ;YES,  IT IS .GT. 999999
 2836  002331' 000000  002124'
 2837  002332' 000000  002111'
 2838  002333' 000000  000074          47160           INR     A
 2839  002334' 001000  000107          47180           MOV     B,A             ;B = DECIMAL POINT COUNT
 2840  002335' 001000  000076          47200           MVI     A,1             ;SET FIXED POINT FLAG, THE EXPONENT IS ZERO
 2841  002336' 000000  000001
 2842                                   47220                                   ; IF WE ARE USING FIXED POINT NOTATION
 2843  002337' 001000  000175          47240   FOUT61  MOV     A               ;E NOTATION, ADD 9 TO ORIGINAL EXPONENT
 2844  002340' 001000  000341          47260           POP     H               ;GET BUFFER POINTER FROM STACK
 2845  002341' 001000  000365          47280           PUSH    PSW             ;SAVE EXPONENT FOR LATER
 2846                                   47300                                   ;CALCULATE THE DIGITS OF THE NUMBER
 2847  002342' 001000  000021          47320           LXI     D,FOUTBL        ;STORE LOC OF LARGEST POWER OF TEN
 2848  002343' 000000  002316'
 2849  002344' 000000  002116'
 2850  002345' 001000  000005          47340   FOUT8:  DCR     B               ;SEE IF IT IS TIME TO PRINT A DECIMAL POINT
 2851  002346' 001000  000066          47360           MVI     M,"."           ;PUT A DECIMAL POINT IN THE BUFFER
 2852  002347' 000000  000056
 2853  002350' 001000  000314          47380           CZ      INXHRT          ;INCREMENT THE BUFFER POINTER IF IT IS TIME
 2854  002351' 000000  001252'
 2855  002352' 000000  002136'
 2856  002353' 001000  000305          47400           PUSH    B               ;SAVE FLAGS
 2857  002354' 001000  000345          47420           PUSH    H               ;SAVE CHARACTER POINTER
```

```
2858  002142' 001000  000325          47440           PUSH    D               ;SAVE POWER OF TEN POINTER
2859  002143' 021000  000315          47460           CALL    MOVMF           ;GET NUMBER IN C,D,E
2860  002146' 000000  001240'
2861  002147' 000000  002136'
2862  002146' 001000  000341          47480           POP     H               ;GET POWER OF TEN POINTER
2863  002147' 001000  000006          47500           MVI     B,"0"-1         ;B = NEXT DIGIT TO BE PRINTED
2864  002150' 000000  000057
2865  002151' 001000  000004          47520   FOUT10: INR     B               ;ADD ONE TO DIGIT
2866  002152' 001000  000173          47540           MOV     A,E             ;SUBTRACT LO
2867  002153' 021007  000226          47560           SUB     M
2868  002154' 001000  000137          47580           MOV     E,A
2869  002155' 001000  000043          47600           INX     H               ;POINT TO NEXT BYTE OF POWER OF TEN
2870  002156' 001000  000172          47620           MOV     A,D             ;SUBTRACT MD
2871  002157' 001000  000236          47640           SBB     M
2872  002160' 001000  000127          47660           MOV     D,A
2873  002161' 001000  000043          47680           INX     H
2874  002162' 001000  000171          47700           MOV     A,C             ;SUBTRACT MO
2875  002163' 001000  000236          47720           SBB     M
2876  002164' 001000  000117          47740           MOV     C,A
2877  002165' 001000  000053          47760           DCX     H               ;POINT TO BEGINNING OF POWER OF TEN
2878  002166' 001000  000053          47780           DCX     H
2879  002167' 001000  000322          47800           JNC     FOUT10          ;SUBTRACT AGAIN IF RESULT WAS POSITIVE
2880  002170' 000000  002151'
2881  002171' 000000  002144'
2882  002172' 001000  000315          47820           CALL    FADDA           ;IT WASN'T, ADD POWER OF TEN BACK IN
2883  002173' 000000  000274'
2884  002174' 000000  002170'
2885  002175' 001000  000043          47840           INX     H               ;INCREMENT POINTER TO NEXT POWER OF TEN
2886  002176' 001000  000315          47860           CALL    MOVFR           ;SAVE C,D,E IN FAC
2887  002177' 000000  001423'
2888  002200' 000000  002173'
2889  002201' 001000  000353          47880           XCHG                    ;GET POWER OF TEN POINTER IN (DE)
2890  002202' 001000  000341          47900           POP     H               ;GET BUFFER POINTER
2891  002203' 001000  000160          47920           MOV     M,B             ;PUT CHARACTER IN BUFFER
2892  002204' 001000  000043          47940           INX     H               ;INCREMENT BUFFER POINTER
2893  002205' 001000  000301          47960           POP     B               ;GET COUNTERS OFF STACK
2894  002206' 001000  000015          47980           DCR     C               ;WAS THAT THE LAST DIGIT?
2895  002207' 001000  000302          48000           JNZ     FOUT8           ;DO MORE IF NOT
2896  002210' 000000  002132'
2897  002211' 000000  002177'
2898  002212' 001000  000005          48020           DCR     B               ;SEE IF DECIMAL POINT GOES AFTER LAST DIGIT
2899  002213' 001000  000312          48040           JZ      FOUT12          ;IT DOES, WE HAVE NO ZEROS TO SUPPRESS
2900  002214' 000000  002232'
2901  002215' 000000  002210'
2902                                   48060           ;SUPPRESS THE TRAILING ZEROS
2903  002216' 001000  000053          48080   FOUT11: DCX     H               ;GO BACK TO LAST CHARACTER
2904  002217' 001000  000176          48100           MOV     A,M             ;GET IT
2905  002220' 001000  000376          48120           CPI     "0"             ;IGNORE TRAILING ZEROS
2906  002221' 000000  000060
2907  002222' 001000  000312          48140           JZ      FOUT11
2908  002223' 000000  002216'
2909  002224' 000000  002214'
2910                                   48160           ;SUPPRESS DECIMAL POINT IF WE HAVE AN INTEGER
```



```
2911  002225' 001000  000376          48180           CPI     "."             ;IGNORE DECIMAL POINT BEFORE TRAILING ZEROS
2912  002226' 000000  000056
2913  002227' 001000  000304          48200           CNZ     INXHRT          ;IF NO DP, MOVE POINTER TO NEXT POSITION
2914  002230' 000000  001252'
2915  002231' 000000  002223'
2916  002232' 001000  000361          48220   FOUT12: POP     PSW             ;GET DECIMAL EXPONENT
2917  002233' 001000  000312          48240           JZ      FOUT17          ;RETURN IF NUMBER WAS IN FIXED POINT FORMAT
2918  002234' 000000  002271'
2919  002235' 000000  002230'
2920                                   48260           ;FLOATING POINT NOTATION -- PUT AN "E" IN THE BUFFER
2921  002236' 001000  000066          48280           MVI     M,"E"           ;PUT AN "E" IN THE BUFFER
2922  002237' 000000  000105
2923  002240' 001000  000043          48300           INX     H               ;PUT SIGN OF EXPONENT IN BUFFER
2924                                   48320           ;PUT IN THE SIGN OF THE EXPONENT
2925  002241' 001000  000066          48340           MVI     M,"+"           ;A PLUS IF POSITIVE
2926  002242' 000000  000053
2927  002243' 001000  000362          48360           JP      FOUT14
2928  002244' 000000  002250'
2929  002245' 000000  002234'
2930  002246' 001000  000066          48380           MVI     M,"-"           ;A MINUS IF NEGATIVE
2931  002247' 000000  000055
2932  002250' 001000  000057          48400           CMA                     ;NEGATE EXPONENT
2933  002251' 001000  000074          48420           INR     A
2934                                   48440           ;CALCULATE THE TWO DIGIT EXPONENT
2935  002252' 001000  000006          48460   FOUT14: MVI     B,"0"-1         ;INITIALIZE TEN'S DIGIT COUNT
2936  002253' 000000  000057
2937  002254' 001000  000004          48480   FOUT15: INR     B               ;INCREMENT DIGIT
2938  002255' 001000  000326          48500           SUI     12              ;SUBTRACT TEN
2939  002256' 000000  002.1C
2940  002257' 001000  000322          48520           JNC     FOUT15          ;DO IT AGAIN IF RESULT WAS POSITIVE
2941  002260' 000000  002254'
2942  002261' 000000  002244'
2943  002262' 001000  000306          48540           ADI     "0"+12          ;ADD BACK IN TEN AND CONVERT TO ASCII
2944  002263' 000000  000072
2945                                   48560           ;PUT THE EXPONENT IN THE BUFFER
2946  002264' 001000  000043          48580           INX     H
2947  002265' 001000  000160          48600           MOV     M,B             ;PUT TEN'S DIGIT OF EXPONENT IN BUFFER
2948  002266' 001000  000043          48620   FOUT19: INX     H               ;WHEN WE JUMP TO HERE, A IS ZERO
2949  002267' 001000  000167          48640           MOV     M,A             ;PUT ONE'S DIGIT IN BUFFER
2950  002270' 001000  000043          48660           INX     H               ;INCREMENT POINTER
2951  002271' 001000  000161          48680   FOUT17: MOV     M,C             ;PUT ZERO AT END OF BUFFER
2952  002272' 001000  000341          48700           POP     H               ;EXIT WITH (HL) POINTING TO STRING
2953  002273' 001000  000311          48720           RET                     ;ALL DONE
2954
2955                                   48760           ;SEE IF FAC .LE. 999999.499
2956  002274' 001000  000.01          48780   FOUTCK: MOVRI   224,164,043,367 ;COMPARE NUMBER WITH CONSTANT
2957  002275' 000000  000164
2958  002276' 000000  000224
2959  002277' 001000  000021
2960  002300' 000000  000367
2961  002301' 000000  000043
2962  002302' 001000  000313          48800           CALL    FCOMP
2963  002303' 000000  001517'
```

```
F4       MAC     23-AUG-64 06:08          FLOATING POINT OUTPUT ROUTINE

 2964  002334' 000000  002266'
 2965  002335' 001000  000341          48820           POP     H               ;GET RETURN ADDRESS OFF STACK
 2966  002336' 001000  000342          48840           JPO     FOUT9           ;NUMBER TOO BIG, DIVIDE BY TEN
 2967  002337' 003000  002266'
 2968  002310' 000000  002303'
 2969  002311' 001000  000351          48860           PCHL                    ;NUMBER OK, RETURN
 2970
 2971                                   48900           ;CONSTANTS FOR FOUT
 2972  002312' 000000  000000          48920   FHALF:  000     ; 1/2
 2973  002313' 000000  000000          48940           000     ;THIS CONSTANT IS ALSO USED BY SQR, SIN, COS
 2974  002314' 000000  000000          48960           000
 2975  002315' 000000  000200          48980           200
 2976                                   49000           ;POWER OF TEN TABLE
 2977  002316' 000000  000240          49020   FOUT8:  240     ; 100000000
 2978  002317' 000000  000206          49040           206
 2979  002320' 000000  000001          49060           001
 2980  002321' 000000  000220          49080           220     ; 10000000
 2981  002322' 000000  000047          49100           047
 2982  002323' 000000  000000          49120           000
 2983  002324' 000000  000350          49140           350     ; 1000000
 2984  002325' 000000  000003          49160           003
 2985  002326' 000000  000000          49180           000
 2986  002327' 000000  000144          49200           144     ; 100000
 2987  002330' 000000  000000          49220           ...
 2988  002331' 000000  000000          49240           000
 2989  002332' 000000  000012          49260           012     ; 10000
 2990  002335' 000000  000000          49280           000
 2991  002334' 000000  000000          49300           000
 2992  002335' 000000  000001          49320           001     ; 1
 2993  002336' 000000  000000          49340           000
 2994  002337' 000000  000000          49360           000>
 2995                                   49380   IFE     LENGTH=2,<
 2996                                   49400           ;OUTPUT THE VALUE IN THE FAC ACCORDING TO THE FORMAT SPECIFICATIONS
 2997                                   ;  IN A,B,C
 2998                                   49420           ;ALL REGISTERS ARE ALTERED
 2999                                   49440           ;THE ORIGINAL CONTENTS OF THE FAC IS LOST
 3000                                   49460
 3001                                   49480           ;THE FORMAT IS SPECIFIED IN A, B AND C AS FOLLOWS:
 3002                                   49500           ;THE BITS OF A MEAN THE FOLLOWING:
 3003                                   49520   ;BIT 7  0 MEANS FREE FORMAT OUTPUT, I.E. THE OTHER BITS OF A MUST BE ZERO,
 3004                                   49540   ;       TRAILING ZEROS ARE SUPPRESSED, A NUMBER IS PRINTED IN FIXED OR FLOATING
 3005
 3006                                   49560   ;       POINT NOTATION ACCORDING TO ITS MAGNITUDE, THE NUMBER IS LEFT
 3007                                   49580   ;       JUSTIFIED IN ITS FIELD, B AND C ARE IGNORED,
 3008                                   49600   ;       1 MEANS FIXED FORMAT OUTPUT, I.E. THE OTHER BITS OF A ARE CHECKED FOR
 3009                                   49620   ;       FORMATTING INFORMATION, THE NUMBER IS RIGHT JUSTIFIED IN ITS FIELD,
 3010                                   49640   ;       TRAILING ZEROS ARE NOT SUPPRESSED.  THIS IS USED FOR PRINT USING,
 3011                                   49660   ;BIT 6  1 MEANS GROUP THE DIGITS IN THE INTEGER PART OF THE NUMBER INTO GROUPS
 3012                                   49680   ;       OF THREE AND SEPARATE THE GROUPS BY COMMAS
 3013                                   49700   ;       0 MEANS DON'T PRINT THE NUMBER WITH COMMAS
 3014                                   49720   ;BIT 5  1 MEANS FILL THE LEADING SPACES IN THE FIELD WITH ASTERISKS ("*")
 3015                                   49740   ;BIT 4  1 MEANS OUTPUT THE NUMBER WITH A FLOATING DOLLAR SIGN ("$")
 3016                                   49760   ;BIT 3  1 MEANS PRINT THE SIGN OF A POSITIVE NUMBER AS A PLUS SIGN ("+")
```



```
F4       MAC     23-AUG-64 06:08          FLOATING POINT OUTPUT ROUTINE

 3017                                   49800   ;       INSTEAD OF A SPACE
 3018                                   49820   ;BIT 2  1 MEANS PRINT THE SIGN OF THE NUMBER AFTER THE NUMBER
 3019                                   49840   ;BIT 1  UNUSED
 3020                                   49860   ;BIT 0  1 MEANS PRINT THE NUMBER IN FLOATING POINT NOTATION I.E. "E NOTATION"
 3021                                   49880   ;       IF THIS BIT IS ON, THE COMMA SPECIFICATION (BIT 6) IS IGNORED,
 3022                                   49900   ;       0 MEANS PRINT THE NUMBER IN FIXED POINT NOTATION.  NUMBER >1E16
 3023                                   49920   ;       CANNOT BE PRINTED IN FIXED POINT NOTATION.
 3024                                   49940
 3025                                   49960           ;B AND C TELL HOW BIG THE FIELD IS:
 3026                                   49980   ;B  =   THE NUMBER OF PLACES IN THE FIELD TO THE LEFT OF THE DECIMAL POINT
 3027                                   50000   ;       (B DOES NOT INCLUDE THE DECIMAL POINT)
 3028                                   50020   ;C  =   THE NUMBER OF PLACES IN THE FIELD TO THE RIGHT OF THE DECIMAL POINT
 3029                                   50040   ;       (C INCLUDES THE DECIMAL POINT)
 3030                                   50060   ;       B AND C DONOT INCLUDE THE 4 POSITIONS FOR THE EXPONENT IF BIT 0 IS ON
 3031                                   50080           ;FOUT ASSUMES B+C .LE. 24 (DECIMAL)
 3032                                   50100
 3033                                   50120           ;ENTRY TO PRINT THE FAC IN FREE FORMAT
 3034  002340  006200          FOUT:   XRA     A               ;SET FORMAT FLAGS TO FREE FORMATED OUTPUT
 3035                                   50160           ;ENTRY TO PRINT THE FAC USING THE FORMAT SPECIFICATIONS IN A, B AND C
 3036  002341  315000  000000  PUFOUT: CALL    FOUINI          ;SAVE THE FORMAT SPECIFICATION IN A AND PUT
 3037                                   50200                   ;A SPACE FOR POSITIVE NUMBERS IN THE BUFFER
 3038  002344  325200          PUSH    B               ;SAVE THE FIELD LENGTH SPECIFICATIONS
 3039  002345  346200          ANI     10              ;CHECK IF POSITIVE NUMBERS GET A PLUS SIGN
 3040  002347  312350  000000  JZ      FOUT1           ;THEY DON'T
 3041  002352  366255          MVI     M,"+"           ;THEY DO, PUT IN A PLUS SIGN
 3042  002354  315300  000000  FOUT1: LDA     VALTYP          ;SEE WHAT KIND OF A VALUE WE HAVE
 3043  002357  327320          MOV     B,A             ;SAVE IT
 3044  002360  315340  000000          XCHG            ;SAVE BUFFER POINTER
 3045  002363  315360  000000          CALL    VSIGN           ;GET THE SIGN OF THE FAC
 3046  002366  315340  000000          XCHG            ;PUT THE BUFFER POINTER BACK IN (H,L)
 3047  002371  327400          MOV     A,B             ;GET THE VALTYP BACK
 3048  002372  362420          JP      FOUT2           ;IF WE HAVE A NEGATIVE NUMBER, NEGATE IT
 3049  002374  366255          MVI     M,"-"           ; AND PUT A MINUS SIGN IN THE BUFFER
 3050  002376  325400          PUSH    H               ;SAVE THE BUFFER POINTER
 3051  002401  315400  000000          CALL    VNEG            ;NEGATE THE NUMBER
 3052  002404  325500          POP     H               ;GET THE BUFFER POINTER BACK
 3053  002405  043520  000000  FOUT2: INX     H               ;POINT TO WHERE THE NEXT CHARACTER GOES
 3054  002410  315340  000000          LDA     TEMP3           ;GET THE FORMAT SPECIFICATION
 3055  002413  327360          MOV     D,A             ;SAVE IT FOR LATER
 3056  002414  307500          RAL                     ;PUT THE FREE FORMAT OR NOT BIT IN THE CARRY
 3057  002415  315340  000000          LDA     VALTYP          ;GET THE VALTYP, VNEG COULD HAVE CHANGED THIS
 3058                                   50620           ; SINCE -32768 IS INT AND 32768 IS SNG,
 3059                                   50640           ; SO B IS NOT ACCURATE
 3060  002422  332520  000000          JC      FOUTFX          ;THE MAN WANTS FIXED FORMATED OUTPUT
 3061                                   50680           ;HERE TO PRINT NUMBERS IN FREE FORMAT
 3062  002425  325200          PUSH    B               ;WE CAN IGNORE THE OLD B AND C
 3063  002426  366260          MVI     M,"0"           ;PUT A ZERO IN THE BUFFER IN CASE THE NUMBER
 3064  002430  312740  000000  JZ      FOUTZR          ; IS ZERO,  IT IS, FINISH IT UP
 3065  002433  376360          CPI     4               ;DECIDE WHAT KIND OF A VALUE WE HAVE
 3066  002435  322750  000000  JNC     FOUFRY          ;WE HAVE A SNG OR DBL
 3067                                   50800           ;HERE TO PRINT AN INTEGER IN FREE FORMAT
 3068  002440  216020  000000          LXI     B,$CODE         ;SET THE DECIMAL POINT COUNT AND COMMA COUNT
 3069                                   50840           ; TO ZERO
```

```
                                        CALL    FOUTC1          ;CONVERT THE INTEGER TO DECIMAL
                                                                ;FALL INTO FOUTZS AND ZERO SUPPRESS THE THING
                                        ;ZERO SUPPRESS THE DIGITS IN FBUFFR
                                        ;ASTERISK FILL AND ZERO SUPPRESS IF NECESSARY
                                        ;SET UP B AND CONDITION CODES IF WE HAVE A TRAILING SIGN
FOUTZS1 LXI     H,FBUFFR-1      ;GET POINTER TO THE SIGN
                MOV     B,M             ;SAVE THE SIGN IN B
                MVI     C," "           ;DEFAULT FILL CHARACTER TO A SPACE
                LDA     TEMP3           ;GET FORMAT SPECS TO SEE IF WE HAVE TO
                MOV     E,A             ; ASTERISK FILL,  SAVE IT
                ANI     40
                JZ      FOTZS1          ;WE DON'T
                MOV     A,B             ;WE DO, SEE IF THE SIGN WAS A SPACE
                CMP     C               ;ZERO FLAG IS SET IF IT WAS
                MVI     C,"*"           ;SET FILL CHARACTER TO AN ASTERISK
                JNZ     FOTZS1          ;SET THE SIGN TO AN ASTERISK IF IT WAS A SPACE
                MOV     B,C             ;B HAS THE SIGN, C THE FILL CHARACTER
FOTZS11 MOV     M,C             ;FILL IN THE ZERO OR THE SIGN
                CHRGET                  ;GET THE NEXT CHARACTER IN THE BUFFER
                                        ;SINCE THERE ARE NO SPACES, "CHRGET" IS
                                        ; EQUIVALENT TO "INX    H"/"MOV A,M"
                CPI     "0"             ;DO WE HAVE A ZERO?
                JZ      FOTZS1          ;YES, SUPPRESS IT
                CPI     36              ;"9"+"," DO WE HAVE A COMMA?
                JZ      FOTZS1          ;YES, SUPPRESS IT
                CPI     "."             ;ARE WE AT THE DECIMAL POINT?
                JNZ     FOTZS2          ;NO, I GUESS NOT
                DCX     H               ;YES, BACK UP AND PUT A ZERO BEFORE IT
                MVI     M,"0"
FOTZS21 MOV     A,E             ;GET THE FORMAT SPECS TO CHECK FOR A FLOATING
                ANI     20              ; DOLLAR SIGN
                JZ      FOTZS3          ;WE DON'T HAVE ONE
                DCX     H               ;WE HAVE ONE, BACK UP AND PUT IN THE DOLLAR
                MVI     M,"$"           ; SIGN
FOTZS31 MOV     A,E             ;DO WE HAVE A TRAILING SIGN?
                ANI     4
                RNZ                     ;YES, RETURN; NOTE THE NON-ZERO FLAG IS SET
FOFXIS1 DCX     H               ;NO, BACK UP ONE AND PUT THE SIGN BACK IN
                                        ;PEOPLE JUMP HERE WHO WANT A "DCX       H" AND
                                        ; DON'T CARE ABOUT H
                MOV     M,B             ;PUT IN THE SIGN
                RET                     ;ALL DONE

                                        ;HERE TO INITIALLY SET UP THE FORMAT SPECS AND PUT IN A SPACE FOR THE
                                        ;SIGN OF A POSITIVE NUMBER
FOUINI: STA     TEMP3           ;SAVE THE FORMAT SPECIFICATION
                LXI     H,FBUFFR-1      ;GET A POINTER INTO FBUFFR
                MVI     M," "           ;PUT IN A SPACE
                RET                     ;ALL DONE

                                        ;HERE TO PRINT A SNG OR DBL IN FREE FORMAT
FOUFRV: PUSH    H               ;SAVE THE BUFFER POINTER
```



```
                JZ      FOUFRS          ;WE HAVE A SNG
                                        ;HERE TO SET UP THE FLAG TO PRINT A DBL IN FREE FORMAT
                MVI     D,20            ;WE HAVE A DBL, SET THE DIGIT COUNT
                DW      1000,001        ;"LXI   B" OVER THE NEXT TWO BYTES
                                        ;HERE TO SET UP THE FLAG TO PRINT A SNG IN FREE FORMAT
FOUFRS: MVI     D,6             ;SET THE DIGIT COUNT
                CALL    FOUTNV          ;NORMALIZE THE FAC SO ALL SIGNIFICANT DIGITS
                                        ; ARE IN THE INTEGER PART
                LXI     B,2*400+SCOVE   ;B = DECIMAL POINT COUNT
                                        ;C = COMMA COUNT
                                        ;SET COMMA COUNT TO ZERO AND DECIMAL POINT
                                        ; COUNT FOR E NOTATION
                ADD     D               ;SEE IF NUMBER SHOULD BE PRINTED IN E NOTATION
                JM      FOFRS1          ;IT SHOULD, IT IS .LT. .1
                INR     D               ;CHECK IF IT IS TOO BG
                CMP     D
                JNC     FOFRS1          ;IT IS TOO BIG, IT IS .GT. 10**D-1
                INR     A               ;IT IS OK FOR FIXED POINT NOTATION
                MOV     B,A             ;SET DECIMAL POINT COUNT
                MVI     A,1             ;SET FIXED POINT FLAG, THE EXPONENT IS ZERO
                                        ; IF WE ARE USING FIXED POINT NOTATION
FOFNS1: DCR     A               ;E NOTATION: ADD D-1 TO ORIGINAL EXPONENT
                POP     H               ;GET THE BUFFER POINTER BACK
                PUSH    PSW             ;SAVE THE EXPONENT FOR LATER
                CALL    FOUTCV          ;CONVERT THE NUMBER TO DECIMAL DIGITS
                                        ;HERE TO SUPPRESS THE TRAILING ZEROS
FOFNS21 DCX     H               ;MOVE BACK TO THE LAST CHARACTER
                MOV     A,M             ;GET IT AND SEE IF IT WAS ZERO
                CPI     "0"
                JZ      FOFNS2          ;IT WAS, CONTINUE SUPPRESSING
                CPI     "."             ;HAVE WE SUPPRESSED ALL THE FRACTIONAL DIGITS?
                CNZ     INCHRT          ;YES, IGNORE THE DECIMAL POINT ALSO
                POP     PSW             ;GET THE EXPONENT BACK
                JZ      FOUTON          ;WE ARE DONE IF WE ARE IN FIXED POINT NOTATION
                                        ;FALL IN AND PUT THE EXPONENT IN THE BUFFER

                                        ;HERE TO PUT THE EXPONENT AND "E" OR "D" IN THE BUFFER
                                        ;THE EXPONENT IS IN A
FOFLON: MOV     B,A             ;SAVE THE EXPONENT
                LDA     VALTYP          ;GET THE VALTYP TO DEIDE IF WE PRINT AN "E"
                CPI     4               ; OR A "D"
                MOV     A,B             ;GET THE EXPONENT BACK
                JZ      FOUCEE          ;WE HAVE TO PRINT AN "E"
                MVI     M,"D"           ;GET THE "D"
                DW      1000,001        ;"LXI   B" OVER THE NEXT TWO BYTES
FOUCEE: MVI     M,"E"           ;GET AN "E"
                INX     H               ;PUT SIGN OF EXPONENT IN BUFFER
                                        ;PUT IN THE SIGN OF THE EXPONENT
                MVI     M,"+"           ;A PLUS IF POSITIVE
                JP      FOUCE1
                MVI     M,"-"           ;A MINUS IF NEGATIVE
                CMA                     ;NEGATE EXPONENT
                INR     A
```

```
                        ;CALCULATE THE TWO DIGIT EXPONENT
FOUCE1: MVI     B,"0"-1         ;INITIALIZE TEN'S DIGIT COUNT
FOUCE2: INR     B               ;INCREMENT DIGIT
        SUI     12              ;SUBTRACT TEN
        JNC     FOUCE2          ;DO IT AGAIN IF RESULT WAS POSITIVE
        ADI     "0"+12          ;ADD BACK IN TEN AND CONVERT TO ASCII
                                ;PUT THE EXPONENT IN THE BUFFER
        INX     H
        MOV     M,B             ;PUT TEN'S DIGIT OF EXPONENT IN BUFFER
        INX     H               ;WHEN WE JUMP TO HERE, A IS ZERO
        MOV     M,A             ;PUT ONE'S DIGIT IN BUFFER
FOUTZR: INX     H               ;INCREMENT POINTER, HERE TO FINISH UP PRINTING
                                ; A FREE FORMAT ZERO
FOUTON: MVI     M,0             ;PUT A ZERO AT THE END OF THE NUMBER
        XCHG                    ;SAVE THE POINTER TO THE END OF THE NUMBER
                                ; IN (DE) FOR FFXFLV
        LXI     H,FBUFFR+1      ;GET A POINTER TO THE BEGINNING
        RET                     ;ALL DONE

                        ;HERE TO PUT A POSSIBLE COMMA COUNT IN C, AND ZERO C IF WE ARE NOT
                        ;USING THE COMMA SPECIFICATION
FOUTCC: MOV     C,A             ;SAVE A POSSIBLE COMMA COUNT
FOUICC: LDA     TEMP3           ;GET THE FORMAT SPECS
        ANI     100             ;LOOK AT THE COMMA BIT
        RNZ                     ;WE ARE USING COMMAS, JUST RETURN
        MOV     C,A             ;WE AREN'T, ZERO THE COMMA COUNT
        RET                     ;ALL DONE

                        ;HERE TO PRINT A NUMBER IN FIXED FORMAT
FOUTFX: CPI     4               ;CHECK WHAT KIND OF VALUE WE HAVE
        MOV     A,D             ;GET THE FORMAT SPECS
        JNC     FOUFXV          ;WE HAVE A BNG OR A DBL
                        ;HERE TO PRINT AN INTEGER IN FIXED FORMAT
        RAR                     ;CHECK IF WE HAVE TO PRINT IT IN FLOATING
        JC      FFXIFL          ; POINT NOTATION
                        ;HERE TO PRINT AN INTEGER IN FIXED FORMAT=FIXED POINT NOTATION
        LXI     B,0+400*3+3*CODE ;SET DECIMAL POINT COUNT TO 0 AND
                                ; COMMA COUNT TO 3
        CALL    FOICC           ;CHECK IF WE DON'T HAVE TO USE THE COMMAS
        POP     D               ;GET THE FIELD LENGTHS
        MOV     A,D             ;SEE IF WE HAVE TO PRINT EXTRA SPACES BECAUSE
        SUI     5               ; THE FIELD IS TOO BIG
        CP      FOUTZER         ;WE DO, PUT IN ZEROS, THEY WILL LATER BE
                                ; CONVERTED TO SPACES OR ASTERISKS BY FOUTZS
        PUSH    D               ;SAVE THE FIELD LENGTHS AGAIN
        CALL    FOUTCI          ;CONVERT THE NUMBER TO DECIMAL DIGITS
        POP     D               ;GET THE FIELD LENGTHS BACK
        DCX     E               ;DO WE NEED A DECIMAL POINT?  HERE ARE
        CZ      FOFXIS          ;WE DON'T, BACKSPACE OVER IT.  AT FOFXIS,
                                ; WE DON'T CARE ABOUT B, WE ONLY WANT THE
                                ; "DCX  M".  LATER ON WE WILL DO A "MVI M,0"
                                ; AND COVER UP WHAT WAS IN B.
        CNZ     FOTZER          ;FOFXIS PRESERVES THE CONDITION CODES
```



```
                                ;IF WE DO HAVE DECIMAL PLACES, FILL THEM UP
                                ; WITH ZEROS
                                ;FALL IN AND FINISH UP THE NUMBER

                        ;HERE TO FINISH UP A FIXED FORMAT NUMBER
FOUTTS: PUSH    H               ;SAVE BUFFER POINTER
        CALL    FOUTZS          ;ZERO SUPPRESS THE NUMBER
        POP     H               ;GET THE BUFFER POINTER BACK
        JZ      FFXIX1          ;CHECK IF WE HAVE A TRAILING SIGN
        MOV     M,B             ;WE DO, PUT THE SIGN IN THE BUFFER
        INX     H               ;INCREMENT THE BUFFER POINTER
FFXIX1: MVI     M,0             ;PUT A ZERO AT THE END OF THE NUMBER

                        ;HERE TO CHECK IF A FIXED FORMAT=FIXED POINT NUMBER OVERFLOWED ITS
                        ;FIELD LENGTH
                        ;D = THE B IN THE FORMAT SPECIFICATION
                        ;THIS ASSUMES THE LOCATION OF THE DECIMAL POINT IS IN TEMP2
        LXI     H,FBUFFR        ;GET A POINTER TO THE BEGINNING
FOUBE1: INX     H               ;INCREMENT POINTER TO THE NEXT CHARACTER
FOUBE5: LDA     TEMP2           ;GET THE LOCATION OF THE DECIMAL POINT
        SUB     L               ;FIGURE OUT HOW MUCH SPACE WE ARE TAKING
        SUB     D               ;IS THIS THE RIGHT AMOUNT OF SPACE TO TAKE?
        RZ                      ;YES, WE ARE DONE, RETURN FROM FOUT
        MOV     A,M             ;NO, WE MUST HAVE TOO MUCH SINCE WE STARTED
                                ; CHECKING FROM THE BEGINNING OF THE BUFFER
                                ; AND THE FIELD MUST BE SMALL ENOUGH TO FIT IN
                                ; THE BUFFER.  GET THE NEXT CHARACTER IN
                                ; THE BUFFER
        CPI     " "             ;IF IT IS A SPACE OR AN ASTERISK, WE CAN
        JZ      FOUBE1          ; IGNORE IT AND MAKE THE FIELD SHORTER WITH
        CPI     "*"             ; NO ILL EFFECTS
        JZ      FOUBE1
        DCX     H               ;MOVE THE POINTER BACK ONE TO READ THE
                                ; CHARACTER WITH CMPGET
        PUSH    H               ;SAVE THE POINTER

                        ;HERE WE SEE IF WE CAN IGNORE THE LEADING ZERO BEFORE A DECIMAL POINT.
                        ;THIS OCCURS IF WE SEE THE FOLLOWING (IN ORDER)
                        ;       A SIGN (EITHER "-" OR "+")      (OPTIONAL)
                        ;       A DOLLAR SIGN                   (OPTIONAL)
                        ;       A ZERO                          (MANDATORY)
                        ;       A DECIMAL POINT                 (MANDATORY)
                        ;       ANOTHER DIGIT                   (MANDATORY)
                        ;IF YOU SEE A LEADING ZERO, IT MUST BE THE ONE BEFORE A DECIMAL POINT
                        ;OR ELSE FOUTZS WOULD HAVE SUPPRESSED IT, SO WE CAN JUST "INX  H"
                        ;OVER THE CHARACTER FOLLOWING THE ZERO, AND NOT CHECK FOR THE
                        ;DECIMAL POINT EXPLICITLY.
FOUBE2: PUSH    PSW             ;PUT THE LAST CHARACTER ON THE STACK.  THE
                                ; ZERO FLAG IS SET.  THE FIRST TIME THE ZERO
                                ; ZERO FLAG IS NOT SET.
        LXI     B,FOUBE2        ;GET THE ADDRESS WE GO TO IF WE SEE A CHARACTER

        PUSH    B               ; WE ARE LOOKING FOR
```

```
3282                                    55280           CHRGET                  ;GET THE NEXT CHARACTER
3283                                    55100           CPI     "."             ;SAVE IT AND GET THE NEXT CHARACTER IF IT IS
3284                                    55120           RZ                      ; A MINUS SIGN, A PLUS SIGN OR A DOLLAR SIGN
3285                                    55140           CPI     "-"
3286                                    55160           RZ
3287                                    55180           CPI     "$"
3288                                    55200           RZ
3289                                    55220           POP     B               ;IT ISN'T, GET THE ADDRESS OFF THE STACK
3290                                    55240           CPI     "0"             ;IS IT A ZERO?
3291                                    55260           JNZ     FOUBE4          ;NO, WE CAN NOT GET RID OF ANOTHER CHARACTER
3292                                    55280           INX     H               ;SKIP OVER THE DECIMAL POINT
3293                                    55300           CHRGET                  ;GET THE NEXT CHARACTER
3294                                    55320           JNC     FOUBE4          ;IT IS NOT A DIGIT, WE CAN'T SHORTEN THE FIELD
3295                                    55340           DCX     H               ;WE CAN;;I  PLINT TO THE DECIMAL POINT
3296                                    55360           DW      1000,001        ;PLXI   B= OVER THE NEXT 2 BYTES
3297                                    55380   FOUBE3: DCX     H               ;POINT BACK ONE CHARACTER
3298                                    55400           MOV     M,A             ;PUT THE CHARACTER BACK
3299                                    55420
3300                                    55440           ;IF WE CAN GET RID OF THE ZERO, WE PUT THE CHARACTERS ON THE STACK
3301                                    55460           ;BACK INTO THE BUFFER ONE POSITION IN FRONT OF WHERE THEY ORIGINALLY
3302                                    55480           ;WERE.  NOTE THAT THE MAXIMUM NUMBER OF STACK LEVELS THIS USES IS
3303                                    55500           ;THREE -- ONE FOR THE LAST ENTRY FLAG, ONE FOR A POSSIBLE SIGN,
3304                                    55520           ;AND ONE FOR A POSSIBLE DOLLAR SIGN.  WE DON'T HAVE TO WORRY ABOUT
3305                                    55540           ;THE FIRST CHARACTER BEING IN THE BUFFER TWICE BECAUSE THE POINTER
3306                                    55560           ;WHEN FOUT EXITS WILL BE POINTING TO THE SECOND OCCURANCE.
3307                                    55580           POP     PSW             ;GET THE CHARACTER OFF THE STACK
3308                                    55600           JZ      FOUBE3          ;PUT IT BACK IN THE BUFFER IF IT IS NOT THE
3309                                    55620                                   ; LAST ONE
3310                                    55640           POP     B               ;GET THE BUFFER POINTER OFF THE STACK
3311                                    55660           JMP     FOUBE3          ;SEE IF THE FIELD IS NOW SMALL ENOUGH
3312                                    55680           ;HERE IF THE NUMBER IS TOO BIG FOR THE FIELD
3313                                    55700   FOUBE4: POP     PSW             ;GET THE CHARACTERS OFF THE STACK
3314                                    55720           JZ      FOUBE4          ;LEAVE THE NUMBER IN THE BUFFER ALONE
3315                                    55740           POP     H               ;GET THE POINTER TO THE BEGINNING OF THE
3316                                    55760                                   ; NUMBER MINUS 1
3317                                    55780           MVI     M,"%"           ;PUT IN A PERCENT SIGN TO INDICATE THE NUMBER
3318                                    55800                                   ; WAS TOO LARGE FOR THE FIELD
3319                                    55820           RET                     ;ALL DONE -- RETURN FROM FOUT
3320                                    55840
3321                                    55860           ;HERE TO PRINT A SNG OR DBL IN FIXED FORMAT
3322                                    55880   FOUFXV: PUSH    H               ;SAVE THE BUFFER POINTER
3323                                    55900           RAR                     ;SET FIXED OR FLOATING NOTATION FLAG IN CARRY
3324                                    55920           JC      FFXFLV          ;PRINT THE NUMBER IN E-NOTATION
3325                                    55940           JZ      FFXSFX          ;WE HAVE A SNG
3326                                    55960           ;HERE TO PRINT A DBL IN FIXED FORMAT--FIXED POINT NOTATION
3327                                    55980           LXI     D,FFXOXM        ;GET POINTER TO 1D16
3328                                    56000           CALL    DCOMPD          ;WE CAN'T PRINT A NUMBER .GE. 10^16 IN FIXED
3329                                    56020                                   ; POINT NOTATION
3330                                    56040           LXI     D,16*400+SCODE  ;SET D = NUMBER OF DIGITS TO PRINT FOR A DBL
3331                                    56060                                   ;C = 0 FOR DBL (THIS IS FOR COMMAS)
3332                                    56080           JM      FFXSDO          ;IF THE FAC WAS SMALL ENOUGH, GO PRINT IT
3333                                    56100           ;HERE TO PRINT IN FREE FORMAT WITH A PERCENT SIGN A NUMBER .GE. 10^16
3334                                    56120   FFXSDO1 IRA     A               ;SET THE FORMAT SPECS TO FREE FORMAT
```



```
3335                                    56140           STA     TEMP3
3336                                    56160           POP     H               ;GET THE BUFFER POINTER
3337                                    56180           DCX     H               ;SET IT UP TO JUMP BACK INTO FOUT
3338                                    56200                                   ;NOW (ML)=FBUFFR+1
3339                                    56220           CALL    FOUT1           ;PRINT THE NUMBER IN FREE FORMAT, THE SIGN
3340                                    56240                                   ; IS ALREADY IN THE BUFFER
3341                                    56260           DCX     H               ;POINT IN FRONT OF THE NUMBER
3342                                    56280           MVI     M,"%"           ;PUT IN THE PERCENT SIGN
3343                                    56300           RET                     ;ALL DONE--RETURN FROM FOUT
3344                                    56320           ;HERE TO PRINT A SNG IN FIXED FORMAT--FIXED POINT NOTATION
3345                                    56340   FFXSFX: MOVRI   226,016,033,312 ;GET 1E16, CHECK IF THE NUMBER IS TOO BIG
3346                                    56360           CALL    FCOMP
3347                                    56380           JP      FFXSDO          ;IT IS, PRINT IT IN FREE FORMAT WITH A % SIGN
3348                                    56400           LXI     D,6*400+2+SCODE ;SET UP FLAGS FOR SNG --
3349                                    56420                                   ;D = NUMBER OF DIGITS TO PRINT IN A SNG
3350                                    56440                                   ;C = 2 (THIS IS FOR THE COMMAS)
3351                                    56460           ;HERE TO ACTUALLY PRINT A SNG OR DBL IN E NOTATION
3352                                    56480   FFXSDO: FSIGN                   ;SEE IF WE HAVE ZERO
3353                                    56500           CNZ     FOUTNV          ;IF NOT, NORMALIZE THE NUMBER SO ALL DIGITS TO
3354                                    56520                                   ; BE PRINTED ARE IN THE INTEGER PART
3355                                    56540           POP     H               ;GET THE BUFFER POINTER
3356                                    56560           POP     B               ;GET THE FIELD LENGTH SPECS
3357                                    56580           JN      FFXXV8          ;DO DIFFERENT STUFF IF [redacted]EXPONENT IS NEGATIVE
3358                                    56590
3359                                    56600           ;HERE TO PRINT A NUMBER WITH NO FRACTIONAL DIGITS
3360                                    56620           PUSH    B               ;SAVE THE FIELD LENGTH SPECS AGAIN
3361                                    56640           MOV     C,A             ;SAVE THE EXPONENT
3362                                    56660           MOV     A,B             ;WE HAVE TO PRINT LEADING ZEROS IF THE FIELD
3363                                    56680           SUB     D               ; HAS MORE CHARACTERS THAN THERE ARE DIGITS IN
3364                                    56700           SUB     C               ; THE NUMBER
3365                                    56720           CP      FOUTZER         ;FOUTZR WILL LATER SUPPRESS THEM
3366                                    56740           MOV     A,C             ;SET UP THE COMMA COUNT
3367                                    56760           ADD     E
3368                                    56780           MOV     E,C             ;WE NO LONGER NEED THE NUMBER WE SETUP IN E
3369                                    56800   FFXXV1: SUI     3               ;REDUCE A MOD 3
3370                                    56820           JNC     FFXXV1
3371                                    56840           ADI     3               ;ADD 3 BACK AND ADD IN 2 MORE FOR SCALING
3372                                    56860           CALL    FOUTCC          ;CHECK IF WE HAVE TO USE COMMAS AT ALL
3373                                    56880           MOV     A,E             ;SETUP THE DECIMAL POINT COUNT
3374                                    56900           ADD     D
3375                                    56920           INR     A
3376                                    56940           MOV     B,A             ;IT GOES IN B
3377                                    56960           PUSH    D               ;SAVE SNG OR DBL AND EXPONENT INFORMATION
3378                                    56980           CALL    FOUTCV          ;CONVERT THE NUMBER TO DECIMAL DIGITS
3379                                    57000           POP     D               ;GET THE SNG OR DBL AND EXPONENT INFO BACK
3380                                    57020           ORA     E               ;PUT IN DIGITS AFTER THE NUMBER IF IT
3381                                    57040                                   ; IS BIG ENOUGH, HERE A=0
3382                                    57060           CNZ     FOUTZRC         ;THERE CAN BE COMMAS IN THESE ZEROS
3383                                    57080           POP     D               ;GET THE FIELD LENGTH SPECS
3384                                    57100           ORA     E               ;HERE A=0
3385                                    57120           CNZ     FOUTDP          ;PRINT A DECIMAL POINT IF NECESSARY
3386                                    57140           OCR     A               ;SEE IF WE SHOULD PRINT SOME ZEROS TO FILL IN
3387                                    57160           CP      FOUTZER         ; THE DECIMAL PLACES
```

```
                                        ;GO CHECK THE SIZE, ZERO SUPPRESS, ETC, AND
                                        ; FINISH THE NUMBER
                                        ;HERE TO PRINT A SNG OR DBL THAT HAS FRACTIONAL DIGITS
FFXXV61 MOV     E,A                     ;SAVE THE EXPONENT, WE DON'T NEED WHAT IS IN E
        MOV     A,C                     ;DIVIDE BY TEN THE RIGHT NUMBER OF TIMES SO
        ORA     A                       ; THE RESULT WILL BE ROUNDED CORRECTLY AND
        CNZ     DCRART                  ; HAVE THE CORRECT NUMBER OF SIGNIFICANT
        ADD     E                       ; DIGITS
        PUSH    PSW                     ;SAVE THIS NUMBER FOR LATER
FFXXV21 CM      FINDIV                  ;THIS IS THE DIVIDE LOOP
        JM      FFXXV2
        MOV     A,E                     ;WE HAVE TWO CASES DEPENDING ON WHETHER THE
        ADD     D                       ; THE NUMBER IS .LT. .1 OR NOT
        MOV     A,B
        JM      FFXXV3
                                        ;HERE TO PRINT NUMBERS .GE. .1
        SUB     D                       ;PRINT SOME LEADING ZEROS IF THE FIELD IS
        SUB     E                       ; BIGGER THAN THE NUMBER OF DIGITS WE WILL
        CP      FOTZER                  ; PRINT
        POP     PSW                     ;WE DON'T NEED THE NUMBER WE SAVED BEFORE
        MOV     B,E                     ;GET ALL THE PERTINENT INFO IN B,C
        PUSH    B                       ;SAVE THE EXPONENT AND "C" IN FIELD SPEC
        MOV     A,E                     ;SET UP THE DECIMAL POINT COUNT
        ADD     D
        INR     A
        MOV     B,A
        MOV     A,D                     ;SET UP THE COMMA COUNT
        ANI     2                       ;THESE 2 INSTRUCTIONS MAP 6 TO 4
        ADI     2                       ; AND 16 TO 2
        ADD     E
        CALL    FOUTCC                  ;CHECK IF WE HAVE TO DO THE COMMA THING
        JMP     FFXXV6                  ;CONVERT THE DIGITS AND DO THE TRIMMING UP
                                        ;HERE TO PRINT A NUMBER .LT. .1
FFXXV31 CALL    FOTZER                  ;PUT ALL ZEROS BEFORE THE DECIMAL POINT
        MOV     A,C                     ;SAVE C
        CALL    FOUTDP                  ;PUT IN A DECIMAL POINT
        MOV     C,A                     ;RESTORE C
        POP     PSW                     ;GET THE NUMBER WE SAVED
        JM      FFXXV4                  ;DECIDE HOW MANY ZEROS TO PRINT BETWEEN THE
        INA     A                       ; DECIMAL POINT AND THE FIRST DIGIT WE WILL
        SUB     E                       ; PRINT,  HERE THE FIELD IS BIG ENOUGH TO
        SUB     D                       ; HOLD ALL THE DIGITS
        JMP     FFXXV5                  ;GO PRINT THEM
FFXXV41 MOV     A,C                     ;HERE WE HAD TO DIVIDE BY TEN SO THE FIELD
        SUB     D                       ; IS SMALLER THAN ALL SIGNIFICANT DIGITS IN
        DCR     A                       ; THE NUMBER
FFXXV51 CALL    FOTZER                  ;PRINT THE ZEROS
        MOV     B,E                     ;SAVE THE EXPONENT IN B
        PUSH    B                       ;SAVE EXPONENT AND THE "C" IN THE FIELD SPEC
        MOV     B,A                     ;ZERO THE DECIMAL PLACE COUNT
        MOV     C,A                     ;ZERO THE COMMA COUNT
FFXXV61 CALL    FOUTCV                  ;CONVERT THE NUMBER TO DECIMAL DIGITS
        POP     D                       ;GET THE EXPONENT AND FIELD SPEC BACK
```



```
        DCR     E                       ;CHECK IF WE HAVE TO PRINT ANY ZEROS AFTER
                                        ; THE LAST DIGIT
        JZ      FFXXV7                  ;CHECK IF THERE WERE ANY DECIMAL PLACES AT ALL
        ADD     D                       ;PRINT SOME MORE TRAILING ZEROS
        DCR     A
        CP      FOTZER
        JMP     FOUTTS                  ;FINISH UP THE NUMBER
                                        ;HERE WERE NO DECIMAL PLACES, IGNORE ALL DIGITS AFTER THE DECIMAL
                                        ;POINT
FFXXV71 LHLD    TEMP2                   ;THE END OF THE NUMBER IS WHERE THE DP IS
        JMP     FOUTTS                  ;FINISH UP THE NUMBER

                                        ;HERE TO PRINT AN INTEGER IN FIXED FORMAT--FLOATING POING NOTATION
FFXIFL1 PUSH    H                       ;SAVE THE BUFFER POINTER
        PUSH    D                       ;SAVE THE FORMAT SPECS
        CALL    CONSI                   ;CONVERT THE INTEGER TO A SNG
        POP     D                       ;GET THE FORMAT SPECS BACK
        POP     H                       ;GET THE BUFFER POINTER BACK
        XRA     A                       ;SET FLAGS TO PRINT THE NUMBER AS A SNG
                                        ;FALL INTO FFXFLV

                                        ;HERE TO PRINT A SNG OR DBL IN FIXED FORMAT=FLOTATING POINT NOTATION
FFXFLV1 JZ      FFXSFL                  ;IF WE HAVE A SNG, SET THE RIGHT FLAGS
        MVI     E,20                    ;WE HAVE A DBL, GET HOW MANY DIGITS WE HAVE
        LXI     H,1000,001              ;"LXI   B" OVER THE NEXT TWO BYTES
FFXSFL1 MVI     E,6                     ;WE HAVE A DBL, GET HOW MANY DIGITS WE PRINT
        FSIGN                           ;SEE IF WE HAVE ZERO
        CNZ     FOUTNV                  ;IF NOT, NORMALIZE THE NUMBER SO ALL DIGITS TO
                                        ; BE PRINTED ARE IN THE INTEGER PART
        POP     H                       ;GET THE BUFFER POINTER BACK
        POP     B                       ;GET THE FIELD LENGTH SPECS
        PUSH    PSW                     ;SAVE THE EXPONENT
        MOV     A,C                     ;CALCULATE HOW MANY SIGNIFICANT DIGITS WE MUST
        ORA     A                       ; PRINT
        PUSH    PSW                     ;SAVE THE "C" FIELD SPEC FOR LATER
        CNZ     DCRART
        ADD     B
        MOV     C,A
        MOV     A,D                     ;GET THE FIELD SPEC
        ANI     8                       ;SEE IF THE SIGN IS A TRAILING SIGN
        CPI     1                       ;SET CARRY IF A IS ZERO
        SBB     A                       ;SET D=0 IF WE HAVE A TRAILING SIGN,
        MOV     D,A                     ; D=377 IF WE DO NOT
        ADD     C
        MOV     C,A                     ;SET C=NUMBER OF SIGNIFICANT DIGITS TO PRINT
        SUB     E                       ;IF WE HAVE LESS THAN E, THEN WE MUST GET RID
FFXLV11 CM      FINDIV                  ; OF SOME BY DIVIDING BY TEN AND ROUNDING
        JP      FFXLV1
        PUSH    B                       ;SAVE THE "B" FIELD SPEC AND # OF SIG DIGITS
        MOV     A,B                     ;SET THE DECIMAL PLACE COUNT
        INR     A
        SUB     D                       ;TAKE INTO ACCOUNT IF THE SIGN IS TRAILING
        MOV     B,A
```

```
                        59300           MVI     C,0             ;SET COMMA COUNT TO ZERO, THE COMMA SPEC IS
                        59320           PUSH    D               ; IGNORED.  SAVE TRAILING SIGN INFO
                        59340           CALL    FOUTCV          ;CONVERT THE NUMBER TO DECIMAL DIGITS
                        59360           POP     D               ;GET THE TRAILING SIGN INFO BACK
                        59380           POP     B               ;GET # OF SIG DIGITS AND "B" FIELD SPEC BACK
                        59400           MOV     A,C             ;PRINT TRAILING ZEROS IF THE FIELD LENGTH IS
                        59420           SUB     E               ; LONGER THAN THE NUMBER OF DIGITS WE CAN PRINT
                        59440           CP      FOTZRO          ;THE DECIMAL POINT COULD COME OUT IN HERE
                        59460           POP     PSW             ;GET THE "C" FIELD SPEC BACK
                        59480           CZ      FOFXIS          ;IF C=0, THE LAST THING WAS A DECIMAL POINT,
                        59500                                   ; SO IGNORE IT.  ALL WE CARE ABOUT IS THE
                        59520                                   ; "0CX  M" AND NOT THE "MOV     M,B" AT FOFXIS
                        59540           POP     PSW             ;GET THE EXPONENT BACK
                        59560           ADD     E               ;SCALE IT CORRECTLY
                        59580           SUB     B
                        59600           SUB     D
                        59620           PUSH    B               ;SAVE THE "B" FIELD SPEC
                        59640           CALL    FOFLDN          ;PUT THE EXPONENT IN THE BUFFER
                        59660           XCHG                    ;GET THE POINTER TO THE END IN (HL)
                        59680                                   ; IN CASE WE HAVE A TRAILING SIGN
                        59700           POP     D               ;GET THE "B" FIELD SPEC IN D, PUT ON A [illegible]E
                        59720           JMP     FOUTTS          ; TRAILING SIGN AND WE ARE DONE
                                                                  Pos. blk
                        59740
                        59760           ;NORMALIZE THE NUMBER IN THE FAC SO ALL THE DIGITS ARE IN THE INTEGER
                        59780           ;PART.  RETURN THE BASE 10 EXPONENT IN A
                        59800           ;D,E ARE LEFT UNALTERED
FOUTNV1:                59820           PUSH    D               ;SAVE (DE)
                        59840           LDA     VALTYP          ;GET WHAT KIND OF VALUE WE HAVE
                        59860           CPI     4
                        59880           JNZ     FOUTN0          ;WE HAVE A DBL
                        59900           ;NORMALIZE A SNG
                        59920           XRA     A               ;ZERO THE EXPONENT
                        59940           PUSH    PSW             ;SAVE IT
                        59960           CALL    FOUNSC          ;IS THE FAC TOO BIG OR TOO SMALL?
FOUNS1:                 60000           MVIRI   241,183,117,37B ;GET 99999,9999 TO SEE IF THE FAC IS BIG
                        60020           CALL    FCOMP           ; ENOUGH YET
                        60040           JPO     FOUNS3          ;IT IS, WE ARE DONE
                        60060           POP     PSW             ;IT ISN'T, MULTIPLY BY TEN
                        60080           CALL    FINMLY
                        60082           PUSH    PSW             ;SAVE THE EXPONENT AGAIN
                        60100           JMP     FOUNS1          ;NOW SEE IF IT IS BIG ENOUGH
FOUNS2: POP             60120           PSW             ;THE FAC IS TOO BIG, GET THE EXPONENT
                        60140           CALL    FINDIV          ;DIVIDE IT BY TEN
                        60160           PUSH    PSW             ;SAVE THE EXPONENT AGAIN
                        60180           CALL    FOUNSC          ;SEE IF THE FAC IS SMALL ENOUGH
FOUNS3: POP             60200           PSW             ;WE ARE DONE, GET THE EXPONENT BACK
                        60220           POP     D               ;GET (DE) BACK
                        60240           RET                     ;ALL DONE
                        60260           ;HERE TO SEE IF THE FAC IS SMALL ENOUGH YET
FOUNSC: MOVRI           60280           224,166,043,367 ;GET 999999,499 TO SEE IF THE FAC IS TOO BIG
                        60300           CALL    FCOMP
```



```
                        60320           POP     H               ;GET THE RETURN ADRESS OFF THE STACK
                        60340           JPO     FOUNS2          ;IT IS TOO BIG, MAKE IT SMALLER
                        60360           PCHL                    ;IT IS SMALL ENOUGH, RETURN
                        60380           ;HERE TO NORMALIZE A DBL NUMBER
                        60400   ;
                        60420   PAGE
```

```
                                        00440   $SUBTTL  EXPONENTIATION AND THE SQUARE ROOT FUNCTION
                                        00460   IFF      EXTFNC,<
                                        00480            ;SQUARE ROOT FUNCTION  ===  X=SQR(A)
                                        00500            ;WE FIRST SCALE THE ARGUMENT TO BETWEEN .5 AND 2 BY LOOKING AT THE
                                        00520            ;EXPONENT AND USING SQR(M*2^(2*N))=2^N*SQR(M).  THEN NEWTON'S METHOD
                                        00540            ;IS USED TO COMPUTE SQR(M).  THE EXPONENT IS SAVED TO SCALE THE
                                        00560            ;RESULT AT THE END.
                                        00580            ;NEWTON'S METHOD FOR SQUARE ROOT:
                                        00600            ;  X[0]=A
                                        00620            ;  X[N+1]=(X[N]+A/X[N])/2
                                        00640   SQR:     FSIGN                   ;CHECK FOR ERROR CONDITION
                                        00660            JM       FLERR          ;CAN'T TAKE SQR OF NEGATIVE NUMBER
                                        00680            RZ                      ;  0=SQR(0)
                                        00700            LXI      H,FAC          ;SCALE ARGUMENT TO BETWEEN .5 AND 2
                                        00720            MOV      A,M            ;GET EXPONENT
                                        00740            RAR                     ;GET EXPONENT OF SCALE FACTOR
                                        00760                                    ;USE  SQR(M*2^(2*N))=2^N*SQR(M)
                                        00780            PUSH     PSW            ;SAVE IT
                                        00800            PUSH     H              ;SAVE POINTER TO EXPONENT
                                        00820            MVI      A,100          ;SET EXPONENT OF SCALED DOWN NUMBER
                                        00840            RAL
                                        00860            MOV      M,A            ;REPLACE IT
                                        00880            LXI      H,FBUFFR       ;SAVE A
                                        00900            CALL     MOVMF
                                        00920            MVI      A,4            ;SET ITERATION COUNT
                                        00940   SQR1:    PUSH     PSW            ;SAVE COUNT
                                        00960            CALL     PUSHF          ;SAVE X[N]
                                        00980            LXI      H,FBUFFR       ;COMPUTE A/X[N]
                                        01000            CALL     MOVRM          ;GET A IN THE REGISTERS
                                        01020            CALL     FDIV
                                        01040            POPX
                                        01060            CALL     FADD           ;ADD IN X[N]
                                        01080            LXI      H,FHALF        ;DIVIDE BY 2
                                        01100            CALL     FMULTS
                                        01120            POP      PSW            ;GET COUNT
                                        01140            DCR      A              ;ARE WE DONE?
                                        01160            JNZ      SQR1           ;NO, DO MORE ITERATIONS
                                        01180            POP      H              ;YES, SET EXPONENT OF ANSWER
                                        01200            POP      PSW            ;GET SCALE FACTOR
                                        01220            ADI      300            ;CONVERT TO AN EXPONENT
                                        01240            ADD      M              ;ADD EXPONENT IN
                                        01260            MOV      M,A            ;REPLACE EXPONENT
                                        01280            RET>                    ;ALL DONE

                                        01340   IFN      EXTFNC,<
                                        01360            ;SUBROUTINE FOR FPWR, ATN
002344' 001000  000041                  01380   PSHNEG:  LXI      H,NEG          ;GET THE ADDRESS OF NEG
002341' 000000  001175'
002342' 000000  002307'
002343' 001000  000343                  01400            XTHL                    ;SWITCH RET ADDR AND ADDR OF NEG
002344' 001000  000351                  01420            PCHL                    ;RETURN, THE ADDRESS OF NEG IS ON THE STACK
```



```
                                        01500            ;SQUARE ROOT FUNCTION
                                        01520            ;WE USE SQR(X)=X^.5
002345' 001000  000515                  01540   SQR:     CALL     PUSHF          ;SAVE ARG
002346' 000000  001205'
002347' 000000  002341'
002350' 001000  000401                  01560            LXI      H,FHALF        ;GET 1/2
002351' 000000  002324'
002352' 000000  002346'
002353' 001000  000515                  01580            CALL     MOVFM          ;  SQR(X)=X^.5
002354' 000000  001222'
002355' 000000  002351'
002356' 001000  000501                  01600   FPWR1:   POPR                    ;GET ARG IN REGISTERS, ENTRY TO FPWR IF
002357' 001000  000321                  01620                                    ;  ARGUMENT IS ON STACK.  FALL INTO FPWR
                                        01640
                                        01660            ;EXPONENTIATION    ===    X^Y
                                        01680            ;N.B.  0^0=1
                                        01700            ;FIRST WE CHECK IF Y=0, IF SO, THE RESULT IS 1.
                                        01720            ;NEXT, WE CHECK IF X=0, IF SO, THE RESULT IS 0.
                                        01740            ;THEN WE CHECK IF X IS POSITIVE, IF NOT, WE CHECK THAT Y IS A
                                        01760            ;NEGATIVE INTEGER, AND WHETHER IT IS EVEN OR ODD.  IF Y IS A NEGATIVE
                                        01780            ;INTEGER, WE NEGATE X.  IF NOT, LOG WILL GIVE AN FC ERROR WHEN WE CALL
                                        01800            ;IT.  IF X IS NEGATIVE AND Y IS ODD, WE PUSH THE ADDRESS OF NEG ON THE
                                        01820            ;STACK SO WE WILL RETURN TO IT AND GET A NEGATIVE RESULT.  TO COMPUTE
                                        01840            ;THE RESULT WE USE X^Y=EXP(Y*LOG(X))
002360' 001000  000357                  01860   FPWR:    FSIGN                   ;SEE IF Y IS ZERO
002361' 001000  000512                  01880            JZ       EXP            ;IT IS, RESULT IS ONE
002362' 000000  002425'
002363' 000000  002354'
002364' 001000  000170                  01900            MOV      A,B            ;SEE IF X IS ZERO
002365' 001000  000267                  01920            ORA      A
002366' 001000  000312                  01940            JZ       ZERO0          ;IT IS, RESULT IS ZERO
002367' 000000  000174'
002370' 000000  002366'
002371' 001000  000325                  01960            PUSHR                   ;SAVE X ON STACK
002372' 001000  000505
002373' 001000  000171                  01980            MOV      A,C            ;CHECK THE SIGN OF X
002374' 001000  000366                  02000            ORI      177            ;TURN THE ZERO FLAG OFF
002375' 000000  000177
002376' 001000  000315                  02020            CALL     MOVMF          ;GET Y IN THE REGISTERS
002377' 000000  001240'
002400' 001000  000362                  02040            JP       FPWR1          ;NO PROBLEMS IF X IS POSITIVE
002401' 001000  000562
002402' 000000  002422'
002403' 000000  002377'
002404' 001000  000325                  02060            PUSHR                   ;SAVE Y
002405' 001000  000505
002406' 001000  000315                  02080            CALL     INT            ;SEE IF Y IS AN INTEGER
002407' 000000  001445'
002410' 000000  002406'
002411' 001000  000301                  02100            POPR                    ;GET Y BACK
```

```
3659  002412' 001000  000521
3660  002413' 001000  000365          62120           PUSH    PSW             ;SAVE LO OF INT FOR EVEN AND ODD INFORMATION
3661  002414' 001000  000315          62140           CALL    FCOMP           ;SEE IF WE HAVE AN INTEGER
3662  002415' 000000  001517'
3663  002416' 000000  002407'
3664  002417' 001000  000341          62160           POP     H               ;GET EVEN-ODD INFORMATION
3665  002420' 001000  000174          62180           MOV     A,H             ;PUT EVEN-ODD FLAG IN CARRY
3666  002421' 001000  000037          62200           RAR
3667  002422' 001000  000341          62220   FPWR1$  POP     H               ;GET X BACK IN FAC
3668  002423' 001000  000042          62240           SHLD    FAC+1           ;STORE MO'S
3669  002424' 777777  777777*
3670  002425' 000000  002415'
3671  002426' 001000  000341          62260           POP     H               ;GET LO'S OFF STACK
3672  002427' 001000  000042          62280           SHLD    FACLO           ;STORE THEM IN FAC
3673  002430' 000000  001454*
3674  002431' 000000  002424'
3675  002432' 001020  000334          62300           CC      PSHNEG          ;NEGATE NUMBER AT END IF Y WAS ODD
3676  002433' 000020  002340'
3677  002434' 000000  002430'
3678  002435' 001000  000314          62320           CZ      NEG             ;NEGATE THE NEGATIVE NUMBER
3679  002436' 000000  001175'
3680  002437' 000000  002435'
3681  002440' 001000  000325          62340   FPWR2$  PUSHR                   ;SAVE Y AGAIN
3682  002441' 001000  000305
3683  002442' 001000  000315          62360           CALL    LOG             ;COMPUTE  EXP(Y*LOG(X))
3684  002443' 000000  000421'
3685  002444' 000000  002436'
3686  002445' 001000  000301          62380           POPR                    ;IF X WAS NEGATIVE AND Y NOT AN INTEGER THEN
3687  002446' 001000  000321
3688  002447' 001000  000315          62400           CALL    FMULT>          ; LOG WILL BLOW HIM OUT OF THE WATER
3689  002450' 000000  001517'
3690  002451' 000000  002443'
3691                                   62420   ;       JMP     EXP
3692                                   62440   PAGE
```



```
3693                                   62460   SUBTTL  EXPONENTIAL FUNCTION
3694                                   62480   IFN     EXTFNC,<
3695                                   62500           ;WE FIRST SAVE THE ORIGINAL ARGUMENT AND MULTIPLY THE FAC BY LOG2(E)
3696                                   62520           ;THE RESULT IS USED TO DETERMINE IF WE WILL GET OVERFLOW SINCE
3697                                   62540           ;EXP(X)=2^(X*LOG2(E)) WHERE LOG2(E)=LOG(E) BASE 2, THEN WE SAVE THE
3698                                   62560           ;INTEGER PART OF THIS TO SCALE THE ANSWER AT THE END, SINCE
3699                                   62580           ;2^Y*2^INT(Y)*2^(Y-INT(Y)) AND 2^INT(Y) IS EASY TO COMPUTE,  SO WE
3700                                   62600           ;NOW COMPUTE 2^(X*LOG2(E)-INT(X*LOG2(E))) BY
3701                                   62620           ;P(LN(2)*(INT(X*LOG2(E))+1)-X) WHERE P IS AN APPROXIMATION
3702                                   62640           ;POLYNOMIAL,  THE RESULT IS THEN SCALED BY THE POWER OF 2 WE
3703                                   62660           ;PREVIOUSLY SAVED,
3704  002452' 001000  000315          62680   EXP$    CALL    PUSHF           ;SAVE ARGUMENT
3705  002453' 000000  001205'
3706  002454' 000000  002450'
3707  002455' 001000  000001          62700           MOVRI   201,070,252,073 ;GET LOG(E) BASE 2,   CACULATE:
3708  002456' 000000  000070
3709  002457' 000000  000201
3710  002460' 001000  000021
3711  002461' 000000  000073
3712  002462' 000000  000252
3713  002463' 001000  000315          62720           CALL    FMULT           ;   INT(ARG/LN(2)) = INT(ARG*LOG2(E))
3714  002464' 000000  000551'
3715  002465' 000000  002453'
3716  002466' 001000  000072          62740           LDA     FAC             ;CARRY=0 IF FAC IS TOO BIG
3717  002467' 000000  021446*
3718  002470' 000000  002464'
3719  002471' 001000  000376          62760           CPI     210             ; I.E. IF ABS(FAC) .GE. 128
3720  002472' 000000  000210
3721  002473' 001000  000322          62780           JNC     MLOVEX          ;IT IS TOO BIG
3722  002474' 000000  001073'
3723  002475' 000000  002467'
3724  002476' 001000  000315          62800           CALL    INT             ;IS ARGUMENT TOO BIG?
3725  002477' 000000  001445'
3726  002500' 000000  002474'
3727  002501' 001000  000306          62820           ADI     200             ;CHECK FOR OVERFLOW
3728  002502' 000000  000200
3729  002503' 001000  000306          62840           ADI     2
3730  002504' 000000  000002
3731  002505' 001000  000332          62860           JC      MLOVEX          ;WE HAVE OVERFLOW!!
3732  002506' 000000  001073'
3733  002507' 000000  002477'
3734  002510' 001000  000365          62880           PUSH    PSW             ;SAVE SCALE FACTOR
3735  002511' 001000  000041          62900           LXI     H,ONE           ;ADD ONE TO THE NUMBER
3736  002512' 000000  000400'
3737  002513' 000000  002506'
3738  002514' 001000  000315          62920           CALL    FADDS
3739  002515' 000000  000001'
3740  002516' 000000  002512'
3741  002517' 001000  000315          62940           CALL    MULLN2          ;MULTIPLY BY LN(2)
3742  002520' 000000  000586'
3743  002521' 000000  002517'
3744  002522' 001000  000361          62960           POP     PSW             ;GET SCALE FACTOR OFF STACK
3745  002523' 001000  000301          62980           POPR                    ;GET ARGUMENT
```

```
3746  002524' 001000  000321
3747  002523' 001000  000365          63200          PUSH    PSW             ;PUT SCALE FACTOR BACK ON STACK
3748  002526' 001000  000315          63220          CALL    FSUB            ;SUBTRACT ORIGINAL ARG
3749  002527' 000020  020017'
3750  002530' 000000  002520'
3751  002531' 001000  000315          63040          CALL    NEG
3752  002532' 000000  001175'
3753  002533' 000000  002527'
3754  002534' 001000  000041          63060          LXI     H,EXPCON        ;EVALUATE THE APPROXIMATION POLYNOMIAL
3755  002535' 000000  002552'
3756  002536' 000000  00253C'
3757  002537' 001000  000315          63080          CALL    POLY
3758  002540' 000000  002532'
3759  002541' 000000  002535'
3760  002542' 001000  000021          63100          LXI     D,SCODE         ;MULTIPLY BY 2 * (B-1) INSTEAD OF JUST
3761  002543' 000000  021477*
3762  002544' 000000  002546'
3763  002545' 001000  000301          63120          POP     B               ; ADDING IT TO THE EXPONENT SO FMULT
3764  002546' 001000  000112          63140          MOV     C,D             ; WILL CHECK FOR EXPONENT OVERFLOW
3765  002547' 001000  000303          63160          JMP     FMULT
3766  002550' 000000  000251'
3767  002551' 000000  002543'
3768
3769                                   63200          ;CONSTANTS FOR EXP
3770  002552' 000000  000010          63220  EXPCON: 10      ;DEGREE
3771  002553' 000000  000100          63240          100     ; =.0001413161
3772  002554' 000000  000056          63260          056
3773  002555' 000000  000224          63280          224
3774  002556' 000000  000164          63300          164
3775  002557' 000000  000104          63320          104     ; .001329886
3776  002560' 000000  000117          63340          117
3777  002561' 000000  000056          63360          056
3778  002562' 000000  000167          63380          167
3779  002563' 000000  000156          63400          156     ; =.00830136
3780  002564' 000000  000002          63420          002
3781  002565' 000000  000210          63440          210
3782  002566' 000000  000172          63460          172
3783  002567' 000000  000346          63480          346     ; .04165735
3784  002570' 000000  000040          63500          240
3785  002571' 000000  000052          63520          052
3786  002572' 000000  000174          63540          174
3787  002573' 000000  000120          63560          120     ; =.1666653
3788  002574' 000000  000052          63580          252
3789  002575' 000000  000252          63600          252
3790  002576' 000000  000176          63620          176
3791  002577' 000000  000377          63640          377     ; .4999999
3792  002600' 000000  000377          63660          377
3793  002601' 000000  000177          63680          177
3794  002602' 000000  000177          63700          177
3795  002603' 000000  000000          63720          000     ; =1.0
3796  002604' 000000  000000          63740          000
3797  002605' 000000  000200          63760          200
3798  002606' 000000  000201          63780          201
```



```
3799  002607' 000000  000000          63800          000     ; 1.0
3800  002610' 000000  000000          63820          000
3801  002611' 000000  000000          63840          000
3802  002612' 000000  000201          63860          201>
3803                                   63880  PAGE
```

```
                                        64900   SUBTTL  POLYNOMIAL EVALUATOR AND THE RANDOM NUMBER GENERATOR
                                        65920   IFN     EXTFNC,<
                                        65940           ;EVALUATE P(X=2)*X
                                        65960           ;POINTER TO DEGREE+1 IS IN (HL)
                                        65980           ;THE CONSTANTS FOLLOW THE DEGREE
                                        66000           ;CONSTANTS SHOULD BE STORED IN REVERSE ORDER, FAC HAS X
                                        66020           ;WE COMPUTE:
                                        66040           ; C0*X*C1*X^3+C2*X^5+C3*X^7+...+C(N)*X^(2*N+1)
002615' 001000  000315  66060   POLYX:  CALL    PUSHF           ;SAVE X
002614' 000000  001203'
002615' 000000  002550'
002610' 001000  000021  66080           LXI     D,FMULTT        ;PUT ADDRESS OF FMULTT ON STACK SO WHEN WE
002617' 000000  060515'
002620' 000000  002614'
                                        66100           PUSH    D               ; RETURN WE WILL MULTIPLY BY X
002622' 001000  000343  66120           PUSH    H               ;SAVE CONSTANT POINTER
002623' 001000  000315  66140           CALL    MOVRF           ;SQUARE X
002624' 000000  001240'
002625' 000000  002617'
002620' 001000  000315  66160           CALL    FMULT
002627' 000000  000517'
002630' 000000  002624'
002631' 001000  000341  66180           POP     H               ;GET CONSTANT POINTER
                                        66200                   ;FALL INTO POLY
                                        66220
                                        66240           ;POLYNOMIAL EVALUATOR
                                        66260           ;POINTER TO DEGREE+1 IS IN (HL), IT IS UPDATED
                                        66300           ;THE CONSTANTS FOLLOW THE DEGREE
                                        66320           ;CONSTANTS SHOULD BE STORED IN REVERSE ORDER, FAC HAS X
                                        66340           ;WE COMPUTE:
                                        66360           ; C0+C1*X+C2*X^2+C3*X^3+...+C(N-1)*X^(N-1)+C(N)*X^N
002632' 001000  000315  66380   POLY1:  CALL    PUSHF           ;SAVE X
002633' 000000  001205'
002634' 000000  002627'
002635' 001000  000176  66400           MOV     A,M             ;GET DEGREE
002636' 001000  000043  66420           INX     H               ;INCREMENT POINTER TO FIRST CONSTANT
002637' 001000  000315  66440           CALL    MOVFM           ;MOVE FIRST CONSTANT TO FAC
002640' 000000  001222'
002641' 000000  002633'
002642' 001000  000006  66460           MVI     B,1000,006      ;"MVI   B," OVER NEXT BYTE
002643' 001000  000361  66480   POLY1:  POP     PSW             ;GET DEGREE
002644' 001000  000301  66500           POPH                    ;GET X
002645' 001000  000321
002646' 001000  000275  66520           DCR     A               ;ARE WE DONE?
002647' 001000  000310  66540           RZ                      ;YES, RETURN
002650' 001000  000325  66560           PUSHH                   ;NO, SAVE X
002651' 001000  000305
002652' 001000  000365  66580           PUSH    PSW             ;SAVE DEGREE
002653' 001000  000345  66600           PUSH    H               ;SAVE CONSTANT POINTER
002654' 001000  000315  66620           CALL    FMULT           ;EVALUATE THE POLY, MULTIPLY BY X
002655' 000000  000517'
002656' 000000  002648'
```



```
002657' 001000  000341  66640           POP     H               ;GET LOCATION OF CONSTANTS
002660' 001000  000315  66660           CALL    MOVRM           ;GET CONSTANT
002661' 000000  001240'
002662' 000000  002655'
                                        66680           PUSH    H               ;STORE LOCATION OF CONSTANTS SO FADJ AND FMULT
002663' 001000  000345
002664' 001000  000315  66700           CALL    FADJ            ; WILL NOT SCREW THEM UP, ADD IN CONSTANT
002665' 000000  000025'
002666' 000000  002661'
002667' 001000  000341  66720           POP     H               ;MOVE CONSTANT POINTER TO NEXT CONSTANT
002670' 001000  000303  66740           JMP     POLY1>          ;SEE IF DONE
002671' 000000  002643'
002672' 000000  002665'
                                        66760
                                        66780
                                        66800           ;PSEUDO-RANDOM NUMBER GENERATOR
                                        66820           ;IF ARG=0, THE LAST RANDOM NUMBER GENERATED IS RETURNED
                                        66840           ;IF ARG .LT. 0, A NEW SEQUENCE OF RANDOM NUMBERS IS STARTED
                                        66860           ; USING THE ARGUMENT
                                        66880           ;TO FORM THE NEXT RANDOM NUMBER IN THE SEQUENCE, WE MULTIPLY THE
                                        66900           ;PREVIOUS RANDOM NUMBER BY A RANDOM CONSTANT, AND ADD IN ANOTHER
                                        66920           ;RANDOM CONSTANT.  THEN THE MO AND LO BYTES ARE SWITCHED, THE
                                        66940           ;EXPONENT IS PUT WHERE IT WILL BE SHIFTED IN BY NORMAL, AND THE
                                        66960           ;EXPONENT IN THE FAC SET TO 200 SO THE RESULT WILL BE LESS THAN 1.
                                        66980           ;THIS IS THEN NORMALIZED AND SAVED FOR THE NEXT TIME.
                                        67000           ;THE MO AND LO BYTES WERE SWITCHED SO WE HAVE A RANDOM CHANCE OF
                                        67020           ;GETTING A NUMBER LESS THAN OR GREATER THAN .5
002673'                                 67040   RND$:
                                        67060   IFN     LENGTH=2,<
002673' 001000  000551  67080           FSIGN>          ;GET SIGN OF ARG
                                        67100   IFE     LENGTH=2,<
                                        67120           CALL    VSIGN           ;GET THE SIGN OF THE ARG
                                        67140           PUSH    PSW             ;SAVE THE SIGN
                                        67160           CM      FNCSNG          ;IF IT IS NEGATIVE, FORCE IT TO BE A SNG
                                        67180           MVI     A,4             ; SINCE WE WILL USE IT
                                        67200           STA     VALTYP          ;MAKE SURE THE RESULT IS "SINGLE PRECISION"
                                        67220           POP     PSW>            ;GET THE SIGN BACK
002674' 001000  000372  67240           JM      RND1            ;START NEW SEQUENCE IF NEGATIVE
002675' 000000  002725'
002676' 000000  002671'
002677' 001000  000341  67260           LXI     H,RNDX          ;GET LAST NUMBER GENERATED
002700' 000000  002752'
002701' 000000  002675'
002702' 001000  000315  67280           CALL    MOVFM
002703' 000000  001222'
002704' 000000  002702'
002705' 001000  000310  67300           RZ                      ;RETURN LAST NUMBER GENERATED IF ZERO
                                        67320   IFE     EXTFNC,<
                                        67340           CALL    FMULTS>         ;MULTIPLY BY CONSTANT A
                                        67360   IFN     EXTFNC,<
002706' 001000  000315  67380           CALL    MOVRM
002707' 000000  001243'
002710' 000000  002705'
```

F4 MAC 23-AUG-64 06:08 POLYNOMIAL EVALUATOR AND THE RANDOM NUMBER GENERATOR

```
3910  002711' 001000  000315          65400            CALL    FMULT>
3911  002712' 000000  000051'
3912  002713' 000000  002707'
3913  002714' 001000  000001          65020            MOVMI   150,050,261,106 ;ADD IN CONSTANT OF ORDER 2^(-24)
3914  002715' 000000  000050
3915  002716' 000000  000150
3916  002717' 001000  000062
3917  002720' 000000  000106
3918  002721' 000000  000261
3919  002722' 001000  000315          65440            CALL    FADL
3920  002723' 000000  000025'
3921  002724' 000000  002712'
3922  002725' 001000  000315          65460   RND11    CALL    MOVMF            ;SWITCH HO AND LO BYTES,
3923  002726' 000000  001640'
3924  002727' 000000  002723'
3925  002730' 001000  000174          65480            MOV     A,E              ;GET LO
3926  002731' 001000  000131          65500            MOV     E,C              ;PUT HO IN LO BYTE
3927  002732' 001000  000117          65520            MOV     C,A              ;PUT LO IN HO BYTE
3928  002733' 001000  000066          65540            MVI     M,200            ;MAKE RESULT POSITIVE
3929  002734' 000000  000200
3930  002735' 001000  000053          65560            DCX     H                ;GET POINTER TO EXPONENT
3931  002736' 001000  000106          65580            MOV     B,M              ;PUT EXPONENT IN OVERFLOW POSITION
3932  002737' 001000  000066          65600            MVI     M,200            ;SET EXP SO RESULT WILL BE BETWEEN 0 AND 1
3933  002740' 000000  000200
3934  002741' 001000  000315          65620            CALL    NORMAL           ;NORMALIZE THE RESULT
3935  002742' 000000  000146'
3936  002743' 000000  002726'
3937  002744' 001000  000041          65640            LXI     H,RNDX           ;SAVE RANDOM NUMBER GENERATED FOR NEXT
3938  002745' 000000  002752'
3939  002746' 000000  002742'
3940  002747' 001000  000303          65660            JMP     MOVMF            ; TIME
3941  002750' 000000  001654'
3942  002751' 000000  002745'
3943
3944                                   65700            ;CONSTANTS AND STORAGE FOR RND
3945  002752' 000000  000122          65720   RNDX1    122              ;LAST RANDOM NUMBER GENERATED, BETWEEN 0 AND 1
3946  002753' 000000  000307          65740            307
3947  002754' 000000  000117          65760            117
3948  002755' 000000  000000          65780            200
3949  002756' 000000  000172          65800            172              ;RANDOM NUMBER OF ORDER 2^24
3950  002757' 000000  000104          65820            104
3951  002760' 000000  000065          65840            065
3952  002761' 000000  000230          65860            230
3953                                   65880   PAGE
```


F4 MAC 23-AUG-64 06:08 SINE, COSINE AND TANGENT FUNCTIONS

```
3954                                   65900   SUBTTL   SINE, COSINE AND TANGENT FUNCTIONS
3955                                   65920   IFN      EXTFNC,<
3956                                   65940            ;COSINE FUNCTION
3957                                   65960            ;IDEA:  USE COS(X)=SIN(X+PI/2)
3958  002762' 001000  000041          65980   COS:     LXI     H,PI2            ;ADD PI/2 TO FAC
3959  002763' 000000  002767'
3960  002764' 000000  002750'
3961  002765' 001000  000315          66000            CALL    FADD3>
3962  002766' 000000  000005'
3963  002767' 000000  002763'
3964                                   66020                                     ;FALL INTO SIN
3965
3966
3967                                   66080            ;SINE FUNCTION
3968                                   66100            ;IDEA:  USE IDENTITIES TO GET FAC IN QUADRANTS I OR IV
3969                                   66120            ;THE FAC IS DIVIDED BY 2*PI AND THE INTEGER PART IS IGNORED BECAUSE
3970                                   66140            ;SIN(X+2*PI)=SIN(X),  THEN THE ARGUMENT CAN BE COMPARED WITH PI/2 BY
3971                                   66160            ;COMPARING THE RESULT OF THE DIVISION WITH PI/2/(2*PI)=1/4.
3972                                   66180            ;IDENTITIES ARE THEN USED TO GET THE RESULT IN QUADRANTS I OR IV,
3973                                   66200            ;AN APPROXIMATION POLYNOMIAL IS THEN USED TO COMPUTE SIN(X).
3974  002770' 001000  000315          66220   SIN:     CALL    PUSHF            ;DIVIDE FAC BY 2*PI
3975  002771' 000000  001205'
3976  002772' 000000  002766'
3977  002773' 001000  000001          66240            MOVMI   203,111,017,333 ;AFTER DIVIDING BY 2*PI, RESULT IS
3978  002774' 000000  000111
3979  002775' 000000  000203
3980  002776' 001000  000021
3981  002777' 000000  000333
3982  003000' 000000  000017
3983  003001' 001000  000315          66260            CALL    MOVFR            ; BETWEEN 0 AND 1
3984  003002' 000000  001425'
3985  003003' 000000  002771'
3986  003004' 001000  000301          66280            POPR
3987  003005' 001000  000321
3988  003006' 001000  000315          66300            CALL    FDIV
3989  003007' 000000  000655'
3990  003010' 000000  003002'
3991  003011' 001000  000315          66320            CALL    PUSHF            ;DISREGARD INTEGER PART SINCE SIN
3992  003012' 000000  001205'
3993  003013' 000000  003007'
3994  003014' 00.000  000315          66340            CALL    INT              ; IS PERIODIC WITH PERIOD 2*PI
3995  003015' 000000  001445'
3996  003016' 000000  003012'
3997  003017' 001000  000301          66360            POPR
3998  003020' 001000  000321
3999  003021' 001000  000315          66380            CALL    FSUB
4000  003022' 000000  000017'
4001  003023' 000000  003015'
4002                                   66400   IFE      EXTFNC,<
4003                                   66420            LXI     B,177*400+$CODE ;GET 1/4
4004                                   66440            MOV     D,C
4005                                   66460            MOV     E,C
4006                                   66480            CALL    FSUB>
```

```
#A      MAC     23-AUG-64 26:08         SINE, COSINE AND TANGENT FUNCTIONS

                                66500   IFN     EXTFNC,<
003024' 021000  000041          66520           LXI     H,FR4           ;SEE WHAT QUADRANT WE ARE IN
003025' 000000  003674'
003026' 003200  003622'
003027' 021000  000315          66540           CALL    FSUB$>
003030' 030002  000011'
003031' 000000  003625'
003032' 021000  000357          66560           FSIGN
003033' 021000  000067          66580           STC                     ;SET QUADRANT I FLAG
003034' 021000  000362          66600           JP      SIN1            ;FIRST QUADRANT, GET BACK ORIGINAL X
003035' 000000  003644'
003036' 000000  003636'
003037' 021000  000315          66620           CALL    FADUM           ;ADD 1/2
003040' 000000  000000'
003041' 020000  003635'
003042' 021000  000357          66640           FSIGN
003043' 021000  000267          66660           DRA     A               ;CLEAR CARRY
003044' 021000  000365          66680   SIN1:   PUSH    PSW             ;SAVE QUADRANT FLAG
003045' 021000  000364          66700           CP      NEG             ;NEGATE IF IN QUADRANTS I, II OR III
003046' 000200  001175'
003047' 000200  003640'

                                66720   IFE     EXTFNC,<
                                66740           LXI     B,177*400+$CODE ;GET 1/4
                                66760           MOV     D,C
                                66780           MOV     E,C
                                66800           CALL    FADD>
                                66820   IFN     EXTFNC,<
003050' 021000  000041          66840           LXI     H,FR4           ;ADD 1/4,  IN QUADRANTS II, III
003051' 000200  003674'
003052' 000200  003646'

                                66860                                   ; USE THE IDENTITY: SIN(PI-X)=SIN(X)
                                66880                                   ;IN QUADRANT IV, USE THE IDENTITY:
003053' 021000  000315          66900           CALL    FADD$>          ;   SIN(X-2*PI)=SIN(X)
003054' 000000  000003'
003055' 000200  003651'
003056' 021000  000361          66920           POP     PSW             ;GET QUADRANT FLAG
003057' 021000  000324          66940           CNC     NEG             ;NEGATE IF IN QUADRANTS II, III OR IV
003060' 000200  001175'
003061' 000000  003654'

                                66960   IFE     EXTFNC,<                ;EVALUATE APPROXIMATION POLYNOMIAL
                                67000           CALL    PUSHF           ;SAVE X
                                67020           CALL    MOVRF           ;SQUARE X
                                67040           CALL    FMULT
                                67060           CALL    PUSHF           ;SAVE X**2
                                67080           LXI     H,$INCON
                                67100           CALL    MOVFM           ;MOVE FIRST CONSTANT INTO FAC
                                67120           POPR                    ;GET X**2
                                67140           MVI     A,4             ;GET DEGREE
                                67160   POLY1:  PUSH    PSW             ;SAVE DEGREE
                                67180           PUSHR                   ;SAVE X**2
                                67200           PUSH    H               ;SAVE CONSTANT POINTER
                                67220           CALL    FMULT           ;EVALUATE THE POLY, MULTIPLY BY X**2
                                67240           POP     H               ;GET POINTER TO CONSTANTS
```


```
F4      MAC     23-AUG-64 26:08         SINE, COSINE AND TANGENT FUNCTIONS

                                67260           CALL    MOVRM           ;GET CONSTANT
                                67280           PUSH    H               ;SAVE POINTER
                                67300           CALL    FADD            ;ADD IN CONSTANT
                                67320           POP     H               ;MOVE POINTER TO NEXT CONSTANT
                                67340           POPR                    ;GET X**2
                                67360           POP     PSW             ;GET DEGREE
                                67380           DCR     A               ;SEE IF DONE
                                67400           JNZ     POLY1           ;NO, DO NEXT TERM
                                67420           JMP     FMULTI>         ;MULTIPLY BY X AND WE ARE DONE
003062' 021000  000041          67440   IFN     EXTFNC,<
003063' 003200  003104'                 LXI     H,$INCON        ;CALCULATE THE SIN BY EVALUATING
003064' 000000  003062'
003065' 021000  000303          67460           JMP     POLYX>          ;  THE APPROXIMATION POLYNOMIAL
003066' 000000  002613'
003067' 000000  003063'
                                67480           ;CONSTANTS FOR SIN, COS
                                67500   IFN     EXTFNC,<
003070' 020000  000333          67520   PI2:    333     ; PI/2
003071' 000000  000017          67540           017
003072' 000000  000111          67560           111
003073' 020000  000201          67580           201
003074' 000000  002000          67600   FR4:    000     ; 1/4
003075' 000000  000000          67620           000
003076' 000000  000000          67640           000
003077' 000000  000177          67660           177>
003100'                         67680   $INCON:
                                67700   IFN     EXTFNC,<
003100' 000000  000005          67720           5>      ;DEGREE
003101' 000000  000272          67740           272     ; 39,781067
003102' 000000  000327          67760           327
003103' 000000  000036          67780           036
003104' 000000  000206          67800           206
003105' 020000  000144          67820           144     ; -76,57498
003106' 020000  000046          67840           046
003107' 020000  000231          67860           231
003110' 000000  000207          67880           207
003111' 000000  000130          67900           130     ; 81,60223
003112' 000000  000064          67920           064
003113' 000000  000043          67940           043
003114' 000000  000207          67960           207
003115' 000000  000340          67980           340     ; -41,34168
003116' 000000  000133          68000           133
003117' 000000  000245          68020           245
003120' 000000  000206          68040           206
003121' 000000  000332          68060           332     ; 6,283185
003122' 000000  000017          68080           017
003123' 000000  000111          68100           111
003124' 000000  000203          68120           203

                                68140
                                68160   IFN     EXTFNC,<
                                68200           ;TANGENT FUNCTION
```

```
                                    68220           ;TAN(X)=SIN(X)/COS(X)
 003125' 001000  000315             68240   TAN:    CALL    PUSHF           ;SAVE ARG
 003126' 000000  001285'
 003127' 000000  001866'
 003130' 001000  000315             68260           CALL    SIN             ;   TAN(X)=SIN(X)/COS(X)
 003131' 000000  002770'
 003132' 000000  001126'
 003133' 001000  000301             68280           POP     B               ;GET X OFF STACK
 003134' 001000  000341             68300           POP     H               ;PUSHF SMASHES (DE)
 003135' 001000  000315             68320           CALL    PUSHF
 003136' 000000  001285'
 003137' 000000  001131'
 003140' 001000  000353             68340           XCHG                    ;GET LOTS WHERE THEY BELONG
 003141' 001000  000315             68360           CALL    MOVFM
 003142' 000000  001285'
 003143' 000000  001136'
 003144' 001000  000315             68380           CALL    CVS
 003145' 000000  002762'
 003146' 000000  001142'
 003147' 001000  000303             68400           JMP     FDIVT
 003150' 000000  000653'
 003151' 000000  001145'
                                    68420   PAGE
```



```
                                    68440   $SBTTL  ARCTANGENT FUNCTION
                                    68460   IFN     EXTFNC,<
                                    68480           ;IDEA: USE IDENTITIES TO GET ARG BETWEEN 0 AND 1 AND THEN USE AN
                                    68500           ;APPROXIMATION POLYNOMIAL TO COMPUTE ARCTAN(X)
 003156' 001000  000357             68520   ATN:    FSIGN                   ;SEE IF ARG IS NEGATIVE
 003157' 001000  000374             68540           CM      PSHNEG          ;IF ARG IS NEGATIVE, USE:
 003160' 000000  002340'
 003161' 000000  003150'
 003162' 001000  000374             68560           CM      NEG             ;   ARCTAN(X)=-ARCTAN(-X)
 003163' 001000  000172             68580           LDA     FAC             ;SEE IF FAC .GT. 1
 003164' 000000  002467'
 003165' 000000  003157'
 003166' 001000  000376             68600           CPI     201
 003167' 000000  000201
 003170' 001000  000332             68620           JC      ATN2
 003171' 000000  003205'
 003172' 000000  100400+
 003173' 000000  003167'
 003174' 001000  000121             68640           LXI     B,201*400+$CODE ;GET THE CONSTANT 1
 003175' 001000  000131             68660           MOV     D,C
 003176' 001000  000315             68680           MOV     E,C             ;COMPUTE RECIPROCAL TO USE THE IDENTITY:
 003177' 000000  000655'            68700           CALL    FDIV            ;  ARCTAN(X)=PI/2-ARCTAN(1/X)
 003200' 000000  003172'
 003201' 001000  000041             68720           LXI     H,FSUBS         ;PUT FSUBS ON THE STACK SO WE WILL RETURN
 003202' 000000  000011'
 003203' 000000  003177'
 003204' 001000  000345             68740           PUSH    H               ; TO IT AND SUBTRACT THE RESULT FROM PI/2
 003205' 001000  000041             68760   ATN2:   LXI     H,ATNCON        ;EVALUATE APPROXIMATION POLYNOMIAL
 003206' 000000  003217'
 003207' 000000  003202'
 003210' 001000  000315             68780           CALL    POLYX
 003211' 000000  002615'
 003212' 000000  003206'
 003213' 001000  000041             68800           LXI     H,PI2           ;GET POINTER TO PI/2 IN CASE WE HAVE TO
 003214' 000000  003270'
 003215' 000000  003211'
 003216' 001000  000311             68820           RET                     ; SUBTRACT THE RESULT FROM PI/2
                                    68840           ;CONSTANTS FOR ATN
 003217' 000000  000011             68860   ATNCON: 11      ;DEGREE
 003220' 000000  000112             68900           112     ; .002866226
 003221' 000000  000327             68920           327
 003222' 000000  000075             68940           075
 003223' 000000  000170             68960           170
 003224' 000000  000002             68980           002     ; -.01616574
 003225' 000000  000156             69000           156
 003226' 000000  000204             69020           204
 003227' 000000  000173             69040           173
 003230' 000000  000376             69060           376     ; .04290961
```

```
4189  003231' 000000  000381          69080           301
4190  003232' 000000  000057          69100           057
4191  003233' 000000  000174          69120           174
4192  003234' 000000  000164          69140           164     ; =.07528964
4193  003235' 000000  000061          69160           061
4194  003236' 000000  000232          69180           232
4195  003237' 000000  000175          69200           175
4196  003240' 000000  000204          69220           204     ; =.1065626
4197  003241' 000000  000075          69240           075
4198  003242' 000000  000132          69260           132
4199  003243' 000000  000175          69280           175
4200  003244' 000000  000310          69300           310     ; =.142089
4201  003245' 000000  000177          69320           177
4202  003246' 000000  000221          69340           221
4203  003247' 000000  000176          69360           176
4204  003250' 000000  000344          69380           344     ; =.1999355
4205  003251' 000000  000273          69400           273
4206  003252' 000000  000114          69420           114
4207  003253' 000000  000176          69440           176
4208  003254' 000000  000154          69460           154     ; =.3333315
4209  003255' 000000  000252          69480           252
4210  003256' 000000  000252          69500           252
4211  003257' 000000  000177          69520           177
4212  003260' 000000  000000          69540           000     ; 1.0
4213  003261' 000000  000000          69560           000
4214  003262' 000000  000000          69580           000
4215  003263' 000000  000201          69600           201>
4216                                  69620   PAGE
```



```
4217                                  69640   SUBTTL  SYSTEM INITIALIZATION CODE
4218                                  69660   RADIX   10              ;IN ALL NON-MATH PACKAGE CODE
4219                                  69680   ;THIS IS THE SYSTEM INITIALIZATION CODE
4220                                  69700   ;IT SHOULD BE LOADED AT THE END OF THE BASIC
4221                                  69720   ;INTERPRETER
4222
4223                                  69760   INTERNAL        INIT
4224
4225                                  69800   EXTERNAL        CRDO,LINGET,QINLIN,READY,SCRTCH,STROUT,REASON,BLF
4226                                  69820   EXTERNAL        $NERR,OMERR,ILLFUN
4227
4228                  145640          69860   FUNIO==<^D256*^O312>+^O48+$CODE
4229  003264'                         69880   INITSA: BLOCK   10
4230  003276' 001000  000041          69900   INITAT: LXI     H,AUTTXT
4231  003277' 002000  004635'
4232  003300' 000000  003214'
4233  003301' 001000  000315          69920           CALL    STROUT
4234  003302' 002000  001763*
4235  003303' 002000  003277'
4236  003304'                         69940   INIT:
4237                                  69960   IFN     REALIO,<
4238  003304' 001000  000333          69980           IN      1               ;IGNORE GARBAGE CHARACTER IN INTERFACE
4239  003305' 000000  000001
4240  003306' 001000  000333          70000           IN      ^D255           ;SEE WHAT KIND OF I/O HE IS
4241  003307' 002000  000377
4242  003310' 001000  000346          70020           ANI     ^O100           ;IS HE FUNNY TTY?
4243  003311' 000000  000100
4244  003312' 001000  000312          70040           JZ      NOTSIU
4245  003313' 000000  003345'
4246  003314' 000000  003302'
4247  003315' 001000  000041          70060           LXI     H,FUNIO
4248  003316' 000000  145644*
4249  003317' 002000  003313'
4250  003320' 001000  000042          70080           SHLD    CNLCA2##+2
4251  003321' 000000  000004*
4252  003322' 000000  003310'
4253  003323' 001000  000046          70100           MVI     M,^O310
4254  003324' 000000  000310
4255  003325' 001000  000042          70120           SHLD    CNLCA3##+2
4256  003326' 000000  000002*
4257  003327' 000000  003321'
4258                                  70140   IFN     LENGTH,<
4259  003330' 001000  000046          70160           MVI     M,^O300         ;SUBSTITUE ^CNZ
4260  003331' 000000  000304
4261  003332' 001000  000042          70180           SHLD    CNLCA4##+2
4262  003333' 000000  000002*
4263  003334' 000000  003326'
4264                  145602          70200   FUNIO==$CODE+<^D256*^O312>+2
4265  003335' 001000  000041          70220           LXI     H,FUNIO
4266  003336' 000000  145602*
4267  003337' 000000  003335'
4268  003340' 001000  000042          70240           SHLD    CNLCA1##+2
4269  003341' 000000  000002*
```

```
4270  003342' 000000  003336'
4271  003343' 001000  000335          70260   NOTSID: IN      ^D255
4272  003344' 000000  000377
4273  003345' 001000  000346          70280           ANI     ^O40
4274  003346' 000000  000040
4275  003347' 001000  000312          70300           JZ      NOTPID
4276  003350' 000000  003400'
4277  003351' 000000  003341'
4278                  165002          70320   FUNIO==.CODE+<^D256+^O312>+2
4279  003352' 001000  000041          70340           LXI     H,FUNIO
4280  003353' 000000  165002#
4281  003354' 000000  003350'
4282  003355' 001000  000042          70360           SHLD    CNLCA2+2
4283  003356' 000000  000002#
4284  003357' 000000  003355'
4285  003360' 001000  000046          70380           MVI     H,^O310
4286  003361' 000000  000310
4287  003362' 001000  000042          70400           SHLD    CNLCA3+2
4288  003363' 000000  000002#
4289  003364' 000000  003356'
4290                                   70420   IFN     .LENGTH,<
4291  003365' 001000  000046          70440           MVI     H,^O304
4292  003366' 000000  000304
4293  003367' 001000  000042          70460           SHLD    CNLCA4+2
4294  003370' 000000  000002#
4295  003371' 000000  003365'
4296                  145001          70480   FUNIO==.CODE+<^D256+^O312>+1
4297  003372' 001000  000041          70500           LXI     H,FUNIO
4298  003373' 000000  145001#
4299  003374' 000000  003370'
4300  003375' 001000  000042          70520           SHLD    CNLCA1+2
4301  003376' 000000  000002#
4302  003377' 000000  003373'
4303  003400'                          70540   NOTPIO:>
4304  003400' 001000  000041          70560           LXI     H,.CODE+^O65535
4305  003401' 000000  177777#
4306  003402' 000000  003376'
4307  003403' 001000  000042          70580           SHLD    CURLIN##        ;IN CASE OF ERROR MESSAGE
4308  003404' 000000  000000#
4309  003405' 000000  003401'
4310  003406' 001000  000041          70600           LXI     H,TSTACK        ;SET UP TEMP STACK
4311  003407' 000000  004315'
4312  003410' 000000  003404'
4313  003411' 001000  000371          70620           SPHL
4314  003412' 001000  000042          70640           SHLD    STKTOP##
4315  003413' 000000  000000#
4316  003414' 000000  003407'
4317                                   70660   IFN     CNTRL#,<
4318  003415' 001000  000257          70680           XRA     A
4319  003416' 001000  000062          70700           STA     CNTWFL##>
4320  003417' 000000  000000#
4321  003420' 000000  003413'
4322  003421' 001000  000315          70720           CALL    CRDO            ;TYPE A CR
```



```
4323  003422' 000000  000000#
4324  003423' 000000  003417'
4325
4326  003424' 001000  000041          70740   IFN     STRING,<
4327  003425' 000000  000000#         70760           LXI     H,TEMPST##
4328  003426' 000000  003422'
4329  003427' 001000  000042          70780           SHLD    TEMPPT##>
4330  003430' 000000  000000#
4331  003431' 000000  003425'
4332                                   70800   IFN     MEALIO,<
4333  003432' 001000  000041          70820           LXI     H,MEMORY        ;ASK HOW MUCH MEMORY AVAILABLE
4334  003433' 000000  004403'
4335  003434' 000000  003430'
4336  003435' 001000  000315          70840           CALL    STROUT          ;CALL THE STRING PRINTER
4337  003436' 000000  003530#
4338  003437' 000000  003433'
4339  003440' 001000  000315          70860           CALL    GINLIN          ;GET A LINE OF INPUT
4340  003441' 000000  000000#
4341  003442' 000000  003436'
4342  003443' 001000  000327          70880           CHRGET                  ;GET A CHAR
4343  003444' 001000  000376          70900           CPI     "A"
4344  003445' 000000  000101
4345  003446' 001000  000312          70920           JZ      INITAT
4346  003447' 000000  003278'
4347  003450' 000000  003441'
4348  003451' 001000  000267          70940           ORA     A
4349  003452' 001000  000302          70960           JNZ     USEDEF          ;NON ZERO, DONT USE DEFAULT
4350  003453' 000000  003501'
4351  003454' 000000  003447'
4352  003455' 001000  000041          70980           LXI     H,LASTMM
4353  003456' 000000  004457'
4354  003457' 000000  003453'
4355  003460' 001000  000043          71000   LOOPMM: INX     H
4356  003461' 001000  000176          71020           MVI     A,31.
4357  003462' 000000  000037
4358  003463' 001000  000167          71040           MOV     M,A
4359  003464' 001000  000276          71060           CMP     M
4360  003465' 001000  000308          71080           JNZ     USEDEF
4361  003466' 000000  003515'
4362  003467' 000000  003458'
4363  003470' 001000  000075          71100           DCR     A
4364  003471' 001000  000167          71120           MOV     M,A
4365  003472' 001000  000276          71140           CMP     M
4366  003473' 001000  000312          71160           JZ      LOOPMM
4367  003474' 000000  003460'
4368  003475' 000000  003466'
4369  003476' 001000  000303          71180           JMP     USEDEF
4370  003477' 000000  003515'
4371  003500' 000000  003474'
4372  003501' 001000  000041          71200   USEDE9: LXI     H,BUF
4373  003502' 000000  000000#
4374  003503' 000000  003477'
4375  003504' 001000  000315          71220           CALL    LINGET          ;GET DECIMAL AMOUNT OF MEMORY IN [D,E]
```

```
4376  003505' 000200  200$00#
4377  003506' 000000  003502'
4378  003507' 001000  000267          71240           ORA     A
4379  003510' 001200  000302          71260           JNZ     $NERR           ;MAKE SURE HE HAS A TERMINATOR
4380  003511' 000000  000000*
4381  003512' 000000  003505'
4382  003514' 001000  000353          71280           XCHG
4383  003514' 001028  002053          71300           OCX     H
4384  003515' 001000  002053          71320   USE.DEF OCX     H>
4385                                   71340   IFE     REAL1U,<
4386                                   71360           LXI     H,$CODE+16190>
4387  003516' 021000  000343          71380           PUSH    H               ;ALSO SAVE FOR LATER
4388  003517' 021000  000041          71400   TTYW:   LXI     H,TTYWIO
4389  003520' 020000  204127'
4390  003521' 020000  003511'
4391  003522' 021000  000315          71420           CALL    STROUT
4392  003523' 020000  003436*
4393  003524' 020000  003526'
4394  003525' 021000  000315          71440           CALL    GINLIN
4395  003526' 020000  003441*
4396  003527' 020000  065523'
4397  003530' 001000  000327          71460           CHRGET
4398  003531' 021000  000267          71480           ORA     A
4399  003532' 021000  000312          71500           JZ      DFLENT
4400  003533' 020000  003684*
4401  003534' 020000  003526'
4402  003535' 021000  000041          71520           LXI     H,BUF
4403  003536' 020000  003502*
4404  003537' 020000  003533'
4405  003540' 021000  000315          71540           CALL    LINGET
4406  003541' 003200  003505*
4407  003542' 000000  003536'
4408  003543' 001000  000172          71560           MOV     A,D
4409  003544' 001200  000267          71580           ORA     A
4410  003545' 001200  000302          71600           JNZ     TTYW
4411  003546' 003200  003517'
4412  003547' 000000  003541'
4413  003550' 001000  000173          71620           MOV     A,E
4414  003551' 001000  000376          71640           CPI     16
4415  003552' 000000  000026
4416  003553' 001000  000332          71660           JC      TTYW
4417  003554' 000000  003517'
4418  003555' 020000  003546'
4419  003556' 021000  000062          71680           STA     LINPTI##        ;DECLARE LINPTI EXTERNAL
4420  003557' 020000  000000*
4421  003560' 020000  003554'
4422                                   71700   IFN     LENGTH,<
4423  003561' 021000  000062          71720           STA     LINPT2##>
4424  003562' 020000  000000*
4425  003563' 020000  003557'
4426                                   71740   IFN     STRING,<
4427  003564' 021000  000062          71760           STA     LINPT3##>
4428  003565' 020000  000000*
```



```
4429  003566' 000000  003562'
4430  003567' 021000  000326          71780   MORCPS: SUI     16
4431  003570' 000000  000016
4432  003571' 001000  003522          71800           JNC     MORCPS
4433  003572' 000000  003567'
4434  003573' 000000  003565'
4435  003574' 001000  000336          71820           ADI     26
4436  003575' 000000  000034
4437  003576' 001000  000057          71840           CMA
4438  003577' 001000  000074          71860           INR     A
4439  003600' 001000  000203          71880           ADD     E
4440  003601' 001000  000062          71900           STA     LINPT4##
4441  003602' 000000  000000*
4442  003603' 000000  003576'
4443  003604'                          71920   DFLENT:
4444                                   71940   IFN     STRING,<                ;SET UP DEFAULT STRING SPACE
4445  003604' 001000  000021          71960           LXI     D,$CODE+^D65536-^D50+1
4446  003605' 000000  1777174
4447  003606' 000000  003602'
4448  003607' 001000  000341          71980           POP     H
4449  003610' 001000  000042          72000           SHLD    MEMSIZ##        ;SAVE IN REAL MEMORY SIZE
4450  003611' 000000  000000*
4451  003612' 000000  003605'
4452  003613' 001000  000042          72020           SHLD    FRETOP##        ;STRINGS START FROM HERE DOWN
4453  003614' 000000  000000*
4454  003615' 000000  003611'
4455  003616' 001000  000031          72040           DAD     D               ;CALC STRBOT BY SUBTRACTING 200 FROM STKTOP
4456  003617' 001000  000322          72060           JNC     OMERR           ;MUST BE POSITIVE
4457  003620' 000000  000000*
4458  003621' 000000  003614'
4459  003622' 001000  000053          72080           DCX     H               ;ONE LOWER IS STKTOP
4460  003623' 001000  000345          72100           PUSH    H>              ;SAVE IT ON STACK
4461                                   72120   IFE     EXTFNC,<
4462                                   72140
4463                                   72160   ; FUNCTION DELETION ROUTINE FOR 4K MACHINE
4464                                   72180   ; TO ADD A NEW FUNCTION JUST UPDATE TBLOO
4465                                   72200
4466                                   72220   ASKAGN: LXI     H,TBLOO         ;START OF FUNCTION TABLE
4467                                   72240   LOPASK: PUSHM                   ;PUT ON CANDIDATE FOR START OF FREE MEMORY
4468                                   72260           LXI     D,TBOASK        ;DONE ASKING?
4469                                   72280           COMPAR
4470                                   72300           JZ      FINFUN          ;YES, GET FIRST FREE MEM LOC
4471                                   72320                                   ;INTO [D,E] AND QUIT
4472                                   72340           PUSHM                   ;PUT ON MESSAGE LOCATION
4473                                   72360           XTHL                    ;[H,L]=MESSAGE POINTER
4474                                   72380                                   ;POINTER INTO TBLOO GOES ON THE STACK
4475                                   72400           CALL    STROUT          ;PRINT THE MESSAGE
4476                                   72420           CALL    GINLIN          ;SEE WHAT HE HAS TO SAY
4477                                   72440           CHRGET                  ;SEE WHAT THE FIRST CHARACTER OF INPUT WAS
4478                                   72460           POP     H               ;POP OFF POINTER INTO TBLJO
4479                                   72480           CPI     "Y"
4480  003624' 000000  003620'  003624' 72500   FINFUN: PUP     D               ;POP OFF CANDIDATE FOR START OF
4481                                   72520                                   ;FREE MEMORY
```

```
4482                                    72540           JZ      MAVFNS          ;HE WANTS IT SO WE ARE DONE
4483                                    72560           CPI     "N"             ;IF A BAD ANSWER
4484                                    72580           JNZ     ASKAGN          ;MAKE HIM START OVER
4485                                    72600           PUSHH                   ;PUSH ON FUNCTION CALL
4486                                    72620                                   ;LOCATION THAT WE FIX UP
4487                                    72640           XTHL                    ;(H,L)=FIXUP PLACE
4488                                    72660                                   ;POINTER INTO TBLDO GOES ON THE STACK
4489                                    72680           LXI     D,ILLFUN        ;THIS IS WHAT WE STORE
4490                                    72700           MOV     M,E
4491                                    72720           INX     H
4492                                    72740           MOV     M,D
4493                                    72760           POP     H               ;GET TBLOO POINTER
4494                                    72780           JMP     LUPASK>         ;GO ASK AGAIN FOR ANOTHER FUNCTION
4495                                    72800   IFN     EXTFNC,<
4496 003624' 001000  000041            72820   ASKAGN: LXI     H,FNS           ;ASK IF WANTS SIN, COS, ATN.
4497 003625' 000000  000012'
4498 003626' 000000  003626'
4499 003627' 001000  000315            72840           CALL    STROUT          ;THE STRING
4500 003630' 000000  0035234
4501 003631' 000000  003625'
4502 003632' 001000  000313            72860           CALL    QINLIN
4503 003633' 000000  0035260
4504 003634' 000000  003632'
4505 003635' 001000  000527            72880           CHRGET
4506 003636' 001000  000376            72900           CPI     "Y"
4507 003637' 000000  000131
4508 003640' 001000  000021            72920           LXI     D,INITSA        ;ASSUME NOT DELETE ANY FNS
4509 003641' 000000  003264'
4510 003642' 000000  003633'
4511 003643' 001000  000312            72940           JZ      MAVFNS          ;YUP.
4512 003644' 000000  003714'
4513 003645' 000000  003641'
4514 003646' 001000  000376            72960           CPI     "A"
4515 003647' 000000  000101
4516 003650' 001000  000312            72980           JZ      OKCHAR
4517 003651' 000000  003660'
4518 003652' 000000  003644'
4519 003653' 001000  000376            73000           CPI     "N"
4520 003654' 000000  000116
4521 003655' 001000  H00302            73020           JNZ     ASKAGN
4522 003656' 000000  003624'
4523 003657' 000000  003651'
4524 003660' 001000  000041            73040   OKCHAR: LXI     H,ILLFUN        ;MAKE SURE BOMBS IF TRIES TO CALL THEM
4525 003661' 000000  0000004
4526 003662' 000000  003656'
4527 003663' 001000  000021            73060           LXI     D,ATN
4528 003664' 000000  003152'
4529 003665' 000000  003661'
4530 003666' 001000  000042            73080           SHLD    ATNFIX##
4531 003667' 000000  0000004
4532 003670' 000000  003664'
4533 003671' 001000  000376            73100           CPI     "A"             ;DELETE ATN BUT NOT SIN, COS?
4534 003672' 000000  000101
```



```
4535 003673' 001000  000312            73120           JZ      MAVFNS          ;TEST
4536 003674' 000000  003712'
4537 003675' 000000  003667'
4538 003676' 001000  000042            73140           SHLD    COSFIX##
4539 003677' 000000  0000004
4540 003700' 000000  003674'
4541 003701' 001000  000042            73160           SHLD    TANFIX##
4542 003702' 000000  0000004
4543 003703' 000000  003677'
4544 003704' 001000  000042            73180           SHLD    SINFIX##
4545 003705' 000000  0000004
4546 003706' 000000  003702'
4547 003707' 001000  000021            73200           LXI     D,COS>
4548 003710' 000000  002762'
4549 003711' 000000  003705'
4550 003712'                            73220   MAVFNS:                         ;TEXT ALWAYS PRECEDED BY ZERO
4551 003712' 001000  000353            73240           XCHG                    ;PUT BOTTOM OF MEMORY IN (H,L)
4552 003713' 001000  000066            73260           MVI     M,0             ;STORE IT
4553 003714' 000000  000000
4554 003715' 001000  000043            73280           INX     H               ;INCREMENT POINTER
4555 003716' 001000  000042            73300           SHLD    TXTTAB##        ;SAVE BOTTOM OF MEMORY
4556 003717' 000000  0000004
4557 003720' 000000  003710'
4558 003721' 001000  000343            73320           XTHL
4559 003722' 001000  000021            73340           LXI     D,TSTACK
4560 003723' 000000  0004315'
4561 003724' 000000  003717'
4562 003725' 001000  000347            73360           COMPAR
4563 003726' 001000  000332            73380           JC      OMERR
4564 003727' 000000  003620*
4565 003730' 000000  003723'
4566 003731' 001000  000321            73400           POP     D
4567 003732' 001000  000371            73420           SPHL
4568 003733' 001000  0004d4            73440           SHLD    STKTOP
4569 003734' 000000  0034134
4570 003735' 000000  003727'
4571 003736' 001000  000353            73460           XCHG
4572 003737' 001000  000315            73480           CALL    REASON
4573 003740' 000000  0004000
4574 003741' 000000  003734'
4575 003742' 001000  000173            73500           MOV     A,E             ;SUBRACT VARTAB FROM STKTOP
4576 003743' 001000  000225            73520           SUB     L               ;LOW PART
4577 003744' 001000  000157            73540           MOV     L,A             ;
4578 003745' 001000  000172            73560           MOV     A,D             ;HIGH PART
4579 003746' 001000  000c34            73580           SBB     H               ;
4580 003747' 001000  000147            73600           MOV     H,A             ;LEAVE IN (B,C)
4581 003750' 001000  000001            73620           LXI     B,$CODE+65520
4582 003751' 000000  177760*
4583 003752' 000000  003740'
4584 003753' 001000  000011            73640           DAD     B
4585 003754' 001000  000315            73660           CALL    CROO            ;TYPE CRLF
4586 003755' 000000  0034224
4587 003756' 000000  003751'
```

```
 4588  003757' 001000  000315          73680            CALL    LINPRT          ;PRINT # OF BYTES FREE
 4589  003760' 000000  001766'
 4590  003761' 000000  003753'
 4591  003762' 001000  000041          73700            LXI     H,WORDS         ;TYPE THE HEADING
 4592  003763' 000000  004146'
 4593  003764' 000000  003762'
 4594  003765' 001000  000315          73720            CALL    STRPRT          ;"ALTAIR BASIC VERSION-----"
 4595  003766' 000000  003630*
 4596  003767' 000000  003763'
 4597  003770' 001000  000041          73740            LXI     H,STROUT
 4598  003771' 000000  003766*
 4599  003772' 000000  003766'
 4600  003773' 001000  000042          73760            SHLD    REPINI##+1
 4601  003774' 000000  000001+
 4602  003775' 000000  003771'
 4603  003776' 001000  000315          73780            CALL    SCRTCH          ;NOW SET UP EVERYTHING ELSE
 4604  003777' 000000  000000*
 4605  004000' 000000  003774'
 4606                                   73800  IFN      LPTSW,<
 4607                                   73820           MVI     A,4
 4608                                   73840           OUT     2               ;CLEAR THE LPT
 4609                                   73860           XRA     A
 4610                                   73880           STA     PRTFLG##
 4611                                   73900           STA     .PTPOS##>
 4612  004001' 001000  000041          73920  IFE      CONSSW,<LXI     H,READY>
 4613  004002' 000000  0000004
 4614  004003' 000000  003777'
 4615                                   73940  IFN      CONSSW,<                        ;
 4616                                   73960           LXI     H,CONSDC##>             ;
 4617  004004' 001000  000042          73980           SHLD    $COJE+2                 ;
 4618  004005' 000000  000002*                                                         ;
 4619  004006' 000000  004004'                                                         ;
 4620  004007' 001000  000351          74000           PCHL                            ;
 4621                                                                                   ;
 4622                                   74040  IFE      EXTFNC,<                        ;
 4623                                   74060  TBLDO:   ADR(INITSA)                     ;
 4624                                   74080           ADR(FNS)                        ;
 4625                                   74100           ADR(SINFIX)                     ;
 4626                                   74120           ADR(SIN)                        ;
 4627                                   74140           ADR(FNS2)                       ;
 4628                                   74160           ADR(RNDFIX)                     ;
 4629                                   74180           ADR(RND)                        ;
 4630                                   74200           ADR(FNS3)                       ;
 4631                                   74220           ADR(SQRFIX)                     ;
 4632                                   74240           ADR(SQR)                        ;
 4633                                   74260  TBDASK:                  ;END OF ASK TABLE
 4634                                   74280  FNS:     DC"WANT SIN"                    ;
 4635                                   74300           0                               ;
 4636                                   74320  FNS2:    DC"WANT RND"                    ;
 4637                                   74340           0                               ;
 4638                                   74360  FNS3:    DC"WANT SQR"                    ;
 4639                                   74380           0>                              ;
 4640                                   74400  IFN      EXTFNC,<                        ;
```



```
 4641  004010' 000000  000127          74420  FNS:     DC"WANT SIN-COS-TAN-ATN"        ;
 4642  004011' 000000  000101                                                          ;
 4643  004012' 000000  000110                                                          ;
 4644  004013' 000000  000124                                                          ;
 4645  004014' 000000  000040                                                          ;
 4646  004015' 000000  000123                                                          ;
 4647  004016' 000000  000111                                                          ;
 4648  004017' 000000  000116                                                          ;
 4649  004020' 000000  000055                                                          ;
 4650  004021' 000000  000103                                                          ;
 4651  004022' 000000  000117                                                          ;
 4652  004023' 000000  000123                                                          ;
 4653  004024' 000000  000055                                                          ;
 4654  004025' 000000  000124                                                          ;
 4655  004026' 000000  000101                                                          ;
 4656  004027' 000000  000116                                                          ;
 4657  004030' 000000  000055                                                          ;
 4658  004031' 000000  000101                                                          ;
 4659  004032' 000000  000124                                                          ;
 4660  004033' 000000  000116                                                          ;
 4661  004034' 000000  000316                                                          ;
 4662  004035' 000000  000000          74440           0>                              ;
 4663  004036' 000000  000215          74460  AUTTXT:  ACRLF                          ;
 4664  004037' 000000  000012                                                          ;
 4665  004038' 000000  000014          74480           "D10                            ;
 4666  004039' 000000  000127          74500           DC"WRITTEN BY BILL GATES & PAUL ALLEN & MONTE DAVIDOFF,"
 4667  004040' 000000  000122                                                          ;
 4668  004041' 000000  000111                                                          ;
 4669  004042' 000000  000124                                                          ;
 4670  004043' 000000  000124                                                          ;
 4671  004044' 000000  000105                                                          ;
 4672  004045' 000000  000116                                                          ;
 4673  004046' 000000  000040                                                          ;
 4674  004047' 000000  000102                                                          ;
 4675  004050' 000000  000131                                                          ;
 4676  004051' 000000  000040                                                          ;
 4677  004052' 000000  000102                                                          ;
 4678  004053' 000000  000111                                                          ;
 4679  004054' 000000  000114                                                          ;
 4680  004055' 000000  000114                                                          ;
 4681  004056' 000000  000040                                                          ;
 4682  004057' 000000  000107                                                          ;
 4683  004060' 000000  000101                                                          ;
 4684  004061' 000000  000124                                                          ;
 4685  004062' 000000  000105                                                          ;
 4686  004063' 000000  000123                                                          ;
 4687  004064' 000000  000040                                                          ;
 4688  004065' 000000  000040                                                          ;
 4689  004066' 000000  000040                                                          ;
 4690  004067' 000000  000120                                                          ;
 4691  004070' 000000  000101                                                          ;
 4692  004071' 000000  000125                                                          ;
 4693  004072' 000000  000114                                                          ;
```

```
4694  004074' 000000  000L40
4695  004075' 000000  000010
4696  004076' 000000  000114
4697  004077' 000000  000114
4698  004100' 000000  000103
4699  004101' 000000  000116
4700  004102' 000000  000040
4701  004103' 000000  000040
4702  004104' 000000  000040
4703  004105' 000000  000115
4704  004106' 000000  000117
4705  004107' 000000  000116
4706  004110' 000000  000124
4707  004111' 000000  000105
4708  004112' 000000  000040
4709  004113' 000000  000124
4710  004114' 000000  000101
4711  004115' 000000  000120
4712  004116' 000000  000111
4713  004117' 000000  000104
4714  004120' 000000  000117
4715  004121' 000000  000105
4716  004122' 000000  000106
4717  004123' 000000  000056
4718  004124' 000000  000056
4719  004124' 000002  000015          74520           ACRLF
4720  004125' 000000  000012
4721  004126' 000000  000000          74540           0
4722
4723  004127' 000000  000124          74560   TTYWID: DC"TERMINAL WIDTH"
4724  004130' 000000  000105
4725  004131' 000000  000122
4726  004132' 000000  000115
4727  004133' 000000  000111
4728  004134' 000000  000116
4729  004135' 000000  000101
4730  004136' 000000  000114
4731  004137' 000000  000040
4732  004140' 000000  000127
4733  004141' 000000  000111
4734  004142' 000000  000124
4735  004143' 000000  000124
4736  004144' 000000  000110
4737  004144' 000000  000310
4738  004145' 000000  000000          74600           0
4739  004146' 000000  000040          74620   WORDS:  DC" BYTES FREE"
4740  004147' 000000  000102
4741  004150' 000000  000101
4742  004151' 000000  000124
4743  004152' 000000  000105
4744  004153' 000000  000123
4745  004154' 000000  000040
4746  004155' 000000  000106
```



```
4747  004156' 000000  000122
4748  004157' 000000  000105
4749  004160' 000000  000105
4750  004161' 000000  000503
4751  004161' 000000  000015          74640           ACRLF
4752  004162' 000000  000012
4753  004163' 000000  000015          74660           ACRLF
4754  004164' 000000  000012
4755  004165' 000000  000101          74680           DC"ALTAIR BASIC VERSION 3.0"
4756  004166' 000000  000114
4757  004167' 000000  000124
4758  004170' 000000  000101
4759  004171' 000000  000111
4760  004172' 000000  000122
4761  004173' 000000  000040
4762  004174' 000000  000102
4763  004175' 000000  000101
4764  004176' 000000  000123
4765  004177' 000000  000111
4766  004200' 000000  000103
4767  004201' 000000  000040
4768  004202' 000000  000126
4769  004203' 000000  000105
4770  004204' 000000  000122
4771  004205' 000000  000123
4772  004206' 000000  000111
4773  004207' 000000  000117
4774  004210' 000000  000116
4775  004211' 000000  000040
4776  004212' 000000  000063
4777  004213' 000000  000056
4778  004214' 000000  000060
4779  004214' 000000  000060
4780  004215' 000000  000015          74700           ACRLF
4781  004216' 000000  000012
4782                                  74720   IFE     LENGTH,<DC"[FOUR-K VERSION]">
4783  004217' 000000  000133          74740   IFE     LENGTH-1,<DC"[EIGHT-K VERSION]">
4784  004220' 000000  000105
4785  004221' 000000  000111
4786  004222' 000000  000107
4787  004223' 000000  000110
4788  004224' 000000  000124
4789  004225' 000000  000055
4790  004226' 000000  000113
4791  004227' 000000  000040
4792  004230' 000000  000126
4793  004231' 000000  000105
4794  004232' 000000  000122
4795  004233' 000000  000123
4796  004234' 000000  000111
4797  004235' 000000  000117
4798  004236' 000000  000116
4799  004237' 000000  000135
```

```
MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 19-11
FB      MAC     23-AUG-64 06:08          SYSTEM INITIALIZATION CODE

  4800  004235' 000000  000335
  4801                                    74760   IFE     LENGTH-2,<0C"[81G VERSION]">
  4802  004240' 000000  000015            74780           ACRLF
  4803  004241' 000000  000612
  4804  004242' 000000  000000            74800           0
  4805
  4806  004243' 000000  000115            74840   MEMJMVI JC"MEMORY SIZE"
  4807  004244' 000000  000105
  4808  004245' 000000  000115
  4809  004246' 000000  000117
  4810  004247' 000000  000122
  4811  004250' 000000  000131
  4812  004251' 000000  000040
  4813  004252' 000000  000123
  4814  004253' 000000  000111
  4815  004254' 000000  000132
  4816  004255' 000000  000105
  4817  004256' 000000  000305
  4818  004256' 000000  000000            74860           0
  4819  004257'                           74880   LASTWMII        BLOCK   "D30            ;LAST WORD OF SYSTEM CODE+1
  4820  004257'                           74900                   BLOCK   "D30            ;SPACE FOR TEMP STACK
  4821  004315'                           74920   TSTACK::
  4822                                    74940   IFE     LENGTH,<
  4823                                    74960           BLOCK   "D13000>
  4824                                    74980   IFN     LENGTH,<
  4825  004315'                           75000           BLOCK   "D10000>
  4826                  000005'           75020           ,C2**:,P
  4827                                    75040   END

NO ERRORS DETECTED

PROGRAM BREAK IS 027735

9K CORE USED
```

```
MATHPK FOR BASIC MCS 8080  GATES/ALLEN/DAVIDOFF MACRO 47(113) 06:09 27-AUG-75 PAGE 20
FB      MAC     23-AUG-64 06:08          SYMBOL TABLE

A               000007          DMULT           001476' INT     FMULT3          000652'
ABS             001173' INT     DMULT1          001506'         FMULT4          000571'
AFFF            002105          DMULT2          001524'         FMULT5          000607'
ASKAGN          003624'         DSKFUN          000000  SPD     FMULTA          000605'
ATN             003152' INT     DY0ERR          000657' EXT     FMULTB          000000'
ATN2            003205'         E               000003          FMULTT          000515' INT
ATNCON          003217'         ENROR           000272' EXT     FNS             004010'
ATNFIX          003667' EXT     ENRQV           000270' EXT     FUNE            000400' INT
ALTTXT          004035'         EXP             002452' INT     FUUT            002001' INT
B               000000          EXPCON          002552'         FOUT1           002215'
BSERR           001522' EXT     EXTFNC          000001  SPD     FOUT10          002151'
BUF             003536' EXT     FAC             003162' EXT     FOUT11          002216'
C               000001          FACLO           002430' EXT     FOUT12          002232'
CASSW           000001  SPD     FADD            000025' INT     FOUT14          002252'
CNLCA1          000000  EXT     FADD1           000057'         FOUT15          002254'
CNLCA2          000000  EXT     FADD3           000125'         FOUT17          002271'
CNLCA3          000000  EXT     FADDA           000270'         FOUT19          002266'
CNLCA4          000000  EXT     FADDM           000000'         FOUT3           002230'
CNTRFL          003417' EXT     FADDS           000003' INT     FOUT5           002271'
CONSSW          000000  SPD     FADDT           000023' INT     FOUT6           002124'
CONTRW          000001  SPD     FADFLT          000143'         FOUT8           002132'
COS             002762' INT     FBUFFR          002002' EXT     FOUT9           002262'
COSFIX          003677' EXT     FLERR           000423' EXT     FOUT8L          002116'
CRDO            003755' EXT     FCOMP           001317' INT     FOUTCB          002274'
CURLIN          003404' EXT     FCOMP2          001347'         FPWR            002160' INT
D               000002          FCOMP4          001340'         FPWR1           002422'
DCRART          001700'         FCOMP8          001141'         FPWR2           002440'
DFLENT          003624'         FDIV            000655' INT     FPWRT           002356' INT
DIV10           000637'         FDIV1           000720'         FR4             003074'
                                FDIV2           000755'         FRETOP          003614' EXT
                                FDIVA           000733'         FSUB            000017' INT
                                FDIVB           000727'         FSUB1           000011'
                                FDIVC           000723'         FSUBT           000015' INT
                                FDIVD           000736'         FUNID           145001  SPD
                                FDIVT           000653' INT     H               000004
                                FMULF           002312'         HAVFNS          003712'
                                FIN             001531' INT     ICOMPS          001142'
                                FIN1            001545'         ILLFUN          003661' EXT
                                FINC            001555'         INIT            003304' INT
                                FINDIG          001702'         INITAT          003276'
                                FINDP           001635'         INITSA          003264'
                                FINE            001641'         INPRT           001756' INT
                                FINE2           001644'         INRART          001145' INT
                                FINE3           001660'         INT             001445' INT
                                FINEC           001621'         INTXT           001760' EXT
                                FINED6          001742'         INXRRT          001252' INT
                                FINLOG          001731'
                                FINNLI          001673'
                                FINNLL          001672'
                                FLOAT           001156' INT
                                FLOATM          001155' INT
                                FMULT           000517' INT
                                FMULT2          000557'
```

```
L               000085          OKCMAN          003660'         $HIFTR          000334'
LASTWR          004257' INT     OMEMR           003727' EXT     $MRADD          000362'
LENGTH          000001  $PD     OVERR           000267' INT     $IGN            001322' EXT
LINGET          003541' EXT     PIB             003070'         $IGNC           001135' INT
LINPNT          001766' INT     PLUSTK          001615' EXT     $IGNS           001143' INT
LINPT1          003557' EXT     POLY            002632'         $IN             002770' INT
LINPT2          003562' EXT     POLY1           002643'         $IN1            003044'
LINPT3          003565' EXT     POLYX           002613'         $INCON          003100'
LINPT4          003602' EXT     PUPMRT          000630'         $INFIX          003705' EXT
LOG             000421' INT     PUPPRT          001474'         $NERR           003511' EXT
LOGCN2          000484'         PSMNEG          002340'         $QR             002345' INT
LOOPNM          003460'         PSW             000006  $PD     STKTOP          003734' EXT
LPTSW           000000  $PD     PUSHF           001205' INT     STRING          000001  $PD
M               000006  $PD     QINLIN          003653' EXT     STRUU1          001776' EXT
MEMORY          004243'         QINT            001372' INT     STROUT          003771' EXT
MEMSIZ          003611' EXT     QINTA           001436'         TAN             003125' INT
MINUTK          001683' EXT     READY           004002' EXT     TANFIX          003702' EXT
MLOVEX          001073'         REALIU          000001  $PD     TEMPPT          003430' EXT
MONCPB          003567'         REASON          003740' EXT     TEMPST          003425' EXT
MOVE            001257' INT     REPINI          000000  EXT     TOSTACK         004315' INT
MOVE1           001261'         RND             002673' INT     TTYM            003517'
MOVFM           001222' INT     RND1            002725'         TTYMID          004127'
MOVFR           001225' INT     RNDX            002752'         TXTTAB          003717' EXT
MOVMF           001254' INT     ROUND           000233'         UNPACK          001272'
MOVRF           001240' INT     ROUNDA          000255'         USEDE0          003501'
MOVRM           001243' INT     ROUNDB          000234'         USEDEF          003515'
MUL10           001146'         SCRTCH          003777' EXT     WORDS           004146'
MULDIM          000081  $PD     SGN             001147' INT     ZERO            000173' INT
MULDIV          001035'         SHFTR1          000336'         ZERO8           000176'
MULDV1          001076'         SHFTR2          000353'         $LOJE           002543' EXT
MULDV2          001077'         SHFTR3          000356'         .CR             002005' SIN
MULLN2          000506'                                         .M              004005' $PD
NEG             001175' INT
NEGR            000510'
NORM1           000151'
NORM2           000200'
NORM3           000218'
NORMAL          001146'
NOTPIO          003400'

A               214     213     218     231     251     299     301     302     304     306     308     310     312     344
                345     346     347     355     356     372     381     383     384     385     386     390     393     401
                412     416     423     427     472     474     476     478     480     482     492     494     495     496
                498     499     501     502     504     505     507     537     538     544     546     547     549     558
                552     553     555     613     684     709     712     739     740     761     762     764     765     767
                768     770     772     784     827     832     837     866     867     868     869     875     878     879
                882     883     886     887     904     905     906     915     917     918     920     921     923     925
                927     935     960     967     971     977     980     986     993     1000    1029    1030    1037    1068
                1070    1096    1126    1129    1247    1265    1269    1274    1275    1279    1333    1334    1343    1349
                1359    1363    1367    1371    1530    1531    1532    1533    1534    1555    1559    1565    1580    1582
                1605    1614    1622    1624    1695    1698    1714    2342    2343    2344    2346    2402    2404    2416
                2429    2559    2579    2580    2582    2686    2694    2716    2771    2801    2812    2838    2839    2848
                2849    2866    2868    2870    2872    2874    2876    2904    2933    2949    3637    3638    3644    3665
                3839    3848    3925    3927    4023    4310    4340    4356    4358    4363    4364    4378    4398    4408
                4409    4413    4438    4575    4577    4578    4580
ABS             89      1111#
AFFF            4642#   4643#   4644#   4645#   4646#   4647#   4648#   4649#   4650#   4651#   4652#   4653#   4654#   4655#
                4656#   4657#   4658#   4659#   4660#   4661#   4661    4667#   4668#   4669#   4670#   4671#   4672#   4673#
                4674#   4675#   4676#   4677#   4678#   4679#   4680#   4681#   4682#   4683#   4684#   4685#   4686#   4687#
                4688#   4689#   4690#   4691#   4692#   4693#   4694#   4695#   4696#   4697#   4698#   4699#   4700#   4701#
                4702#   4703#   4704#   4705#   4706#   4707#   4708#   4709#   4710#   4711#   4712#   4713#   4714#   4715#
                4716#   4717#   4718#   4719    4724#   4725#   4726#   4727#   4728#   4729#   4730#   4731#   4732#   4733#
                4734#   4735#   4736#   4737#   4737    4740#   4741#   4742#   4743#   4744#   4745#   4746#   4747#   4748#
                4749#   4750#   4751    4756#   4757#   4758#   4759#   4760#   4761#   4762#   4763#   4764#   4765#   4766#
                4767#   4768#   4769#   4770#   4771#   4772#   4773#   4774#   4775#   4776#   4777#   4778#   4779#   4779
                4784#   4785#   4786#   4787#   4788#   4789#   4790#   4791#   4792#   4793#   4794#   4795#   4796#   4797#
                4798#   4799#   4800#   4800    4807#   4808#   4809#   4810#   4811#   4812#   4813#   4814#   4815#   4816#
                4817#   4817
ASKAGN          4496#   4522
ATN             92      4140#   4528
ATN2            4155    4167#
ATNCON          4168    4179#
ATNFIX          4531#   4531
AUTTXT          4231    4663#
B               198     210     212     226     240     300     301     302     345     356     379     390     392     412
                420     497     498     518     527     553     555     620     622     624     628     630     674     692
                695     696     708     771     781     806     864     874     885     886     902     925     927     986
                976     977     979     980     1029    1037    1083    1097    1098    1199    1225    1241    1250    1333
                1359    1530    1561    1570    1695    1710    2342    2417    2580    2582    2583    2602    2717    2823
                2839    2850    2850    2861    2863    2891    2893    2898    2935    2937    2947    3618    3637    3643
                3654    3658    3662    3686    3745    3763    3846    3851    3931    3986    3997    4120    4155    4581
                4584
BSERR           1702#   1712
BUF             42254   4373    4405
C               311     312     326     352     384     386     426     427     454     456     481     482     506     507
                529     530     544     546     684     748     761     783     784     864     867     904     921     923
                935     1096    1200    1223    1275    1279    1343    1349    1355    1363    1531    1585    1624    1696
                2346    2405    2408    2581    2828    2874    2876    2894    2951    3644    3764    3926    3927    4158
                4159
CASSW           [redacted]
CNLEA1          4269#   4269    4281
CNLEA2          4251#   4251    4283
CNLEA3          4250#   4256    4288
```

```
CNLC4A    4264#   4262    4294
CNTWPL    4320#   4320
CONSSW      9#    4612    4615
CONTRW     11#    4317
COS        92    3956#   4130    4548
COSFIX    4539#   4539
CROU      4223#   4323    4586
CURLIN    4306#   4306
D          199     211     641     387     388     352     353     381     383     452     477     478     503     504
           528     529     547     549     623     629     673     695     739     746     747     750     751     772
           782     783     807     868     918     920     936    1085    1221    1232    1246    1248    1367    1532
          1579    1580    2345    2370    2380    2398    2436    2578    2579    2585    2603    2847    2858    2870
          2872    3619    3642    3653    3659    3681    3687    3746    3760    3764    3815    3818    3847    3858
          3987    3998    4158    4428    4445    4455    4508    4527    4547    4559    4566    4578
QCRART    2557#
QFLENT    4400    4443#
QIV10      79#    2426    2798
QMULT      94    1892#
QMULT1    1700#   1716
QMULT2    1700    1714#
QSKPUN      8#
QVOERR    1006#    815
E          305     304     383     391     450     463     473     474     500     501     527     528     550     552
           690     721     771     781     782     869     915     917     937    1219    1371    1533    1581    1622
          2344    2403    2404    2410    2686    2689    2694    2866    2868    3925    3926    4159    4413    4439
          4575
ERROR     100#     466
ERROV     100#     464
EXP        92    3635    3704#
EXPCON    3755    3770#
EXPFNC     3d#      91     191     580     670     996    1008    1013    1021    1026    1028    2592    2597    2667
          3558    3596    3694#   3805    3904    3906    3955    4002    4007    4028    4033    4046    4069    4077
          4087    4111    4137    4461    4495    4622    4640
FAC       106#     216     374     398     414     490     611     943     973    1042    1063    1094    1124    1175
          1202    1263    1346    1603    3669    3717    4148
FACLD     106#     270     707    1170    1197    1211    1233    1610    3673
FADD        85     186     212#    626     653    1039    3864    3920
FADD1     228     242#
FADD3     273     299#
FADDA     275     472#    2883
FADDM     179#    2812    4020
FADDS      86     182#    3739    3962    4040
FADDT      98     210#    2677
FADFLT    315#    1105    1630
FBUFFR    106#    2743
FCERR     106#     606
FCOMP      89    1333#    2783    2963    3662
FCOMP2    1356    1358#
FCOMPD    1354#
FCOMPS    1066#   1339
FDIV       86     632     813#    3989    4161
FDIV1     873#    939     948
FDIV2     895     902#
```

```
FDIVA     829     854#
FDIVB     834     862#
FDIVC     839     876#
FDIVG     871     887#    896     929     935
FDIVT      98     806#    4133
FMALF     100    2972#    3615
FIN        80    2322#
FIN1      2329    2334    2337#
FINC      2348#   2410    2605
FINDIG    2350    2576#
FINDP     2353    2408#
FINE      2360    2400    2415#
FINE2     2418#   2431
FINE3     2422    2430#
FINEC     2367    2394#   2696
FINEDG    2396    2686#
FINLOG     657    2599    2869#
FINMLT    2546#   2791
FINMUL    2819    2545#
FLOAT      86    1083#   2673
FLOATR     88    1093#   2726#
FMULT      88     676#    3689    3714    3765    3824    3855    3911
FMULT2    702     709#
FMULT3    714     781#
FMULT4    735#    774
FMULT5    742     760#
FMULTA    686     752#
FMULTB    690     747#
FMULTT     96     674#    3810
FNS       4497    4641#
FONE       92     586#    635     3736
FOUT       86    2742#
FOUT1     2751    2755#
FOUT10    2865#   2880
FOUT11    2903#   2908
FOUT12    2900    2910#
FOUT14    2928    2935#
FOUT15    2937#   2941
FOUT17    2918    2951#
FOUT19    2739    2948#
FOUT3     2776#   2795
FOUT5     2787    2809#
FOUT6     2831    2836    2843#
FOUT8     2850#   2896
FOUT9     2797#   2967
FOUTBL    2848    2977#
FOUTCB    2774    2804    2956#
FPWR       92    3633#
FPWR1     3631    3667#
FPWR2     3681#
FPWRT      100    3618#
FR4       4089    4035    4082#
FRETOP    4455#   4453
```

```
FSUM      88    200#  3744  4000
FSU8S    192#   638   4012  4164
FSU8T     98    198#
FUNIO   4226#  4248   4264#  4266  4278#  4280  4296#  4298
M        179    251    268   269   282    305   309    343   353   354   380   391   397   413
         422    475    479   489   610    634   640    697   700   701   704   705   706   710
         716    762    764   776   826    831   836    873   879   882   903   924   941   942
         946    972    994   999  1001   1041  1093   1100  1123  1173  1178  1199  1210  1220
        1222   1224   1226  1249  1262   1272  1273   1338  1341  1345  1358  1362  1366  1370
        1374   1375   1547  1555  1565   1576  1602   1692  1700  1705  2336  2365  2366  2392
        2413   2584   2601  2701  2702   2708  2726   2729  2742  2745  2755  2761  2844  2857
        2862   2869   2873  2877  2878   2885  2890   2892  2903  2923  2946  2948  2950  2952
        2965   3600   3612  3664  3665   3667  3671   3735  3754  3819  3826  3840  3853  3857
        3861   3866   3897  3938  3937   3958  4008   4034  4070  4121  4163  4166  4167  4173
        4230   4247   4253  4259  4265   4279  4285   4291  4297  4304  4310  4326  4333  4352
        4355   4372   4383  4384  4387   4388  4402   4448  4459  4460  4496  4524  4554  4579
        4580   4591   4597  4612
HAVFNS  4512   4536   4550#
ICOMPS  1067#
ILLFUN  4226#  4525
INIT    4223   4236#
INITAT  4230#  4346
INIT&A  4229#  4509
INPRT     90   2701#
INRART    90   1070#
INT       89   1602#  3656  3725  3995
INTXT   2703#  2703
INXRRT    90   1226#  2854  2914
L        390    342    354   355   392    496    499   502   505   537   539   678   765   767
         817    875    878   971  1200   4576   4577
LASTWR  4353   4819#
LENGTH    4#     19     29    34    79     93     97    99   101   108   195   197   208   243
         287    289    327   341   433    910    920   941   943   961   972   711   717   719
         736    738    744   754   759    776    780   793   803   912   914   957  1002  1015
        1076   1112   1133  1243  1285   1379   1434  1584  1586  1595  1636  1688  1721  1939
        2338   2443   2547  2551  2562   2577   2607  2675  2679  2714  2722  2733  2741  2995
        3885   3887   4258  4290  4422   4782   4783  4801  4822  4824
LINGET  4225#  4376   4406
LINPRT    90   2713#  4589
LINPT1  4420#  4420
LINPT2  4424#  4424
LINPT3  4426#  4428
LINPT4  4441#  4441
LOG       92   6044   3684
LOGLN2  5908#  641
LOOPMM  4355#  4367
LPT&M     7#   4686
M        283    302    306   310   408    401   420   423   458   472   476   488   492   494
         613    622    709   822   823    827   832   837   945   975   986   993  1044  1097
        1101   1126   1129  1219  1221   1223  1225  1247  1265  1269  1271  1274  1281  1348
        1360   1364   1368  1372  1554   1605  1614  1618  2691  2748  2753  2756  2851  2867
        2871   2875   2891  2904  2921   2925  2938  2947  2949  2951  3839  3928  3931  3932
        4358   4359   4364  4365  4552
MEMORY  4334   4086#
MEMSIZ  4450#  4450
MINUTK   106#  2372
MLOVEX   997#  3722   3732
MORCPS  4438#  4433
MOVE      88   1241#
MOVE1   1246#  1252
MOVFM     90   1187#  3618  3842  3901
MOVFR     90    220    238   429   801   1195#  2820  2847  3984  4127
MOVHF     90   1232#  3941
MOVHF     90   1027   1210#  1549  2860  3648  3821  3923
MOVHM     90    183    193  1188  1219#  3859  3908
MUL10   1026#  2549   2587
MULDIM    31#    93   1088
MULDIV   681    820    966#
MULOV1   984   1000#
MULOV2   969   1001#
MULLN2   659#  3742
NEG       90    201   1123#  2439  2763  3601  3679  3752  4026  4044  4145
NEGN     316    489#  1573
NORM1    345#   362
NORM2   1379#   388
NORM3    349    387#
NORMAL   528#   698   3935
NOTPIO  4276   4303#
NOTSIO  4245   4271#
OKCHAR  4517   4524#
OMERR   4226#  4457   4564
OVERR     89    285    463#   951  1020  1035  1047
PI2     3959   4078#  4174
PLUSTK   106#  2386
POLY    3758   3836#
POLY1   3885#  3866
POLYX    644   3812#  4074  4171
POPMRT   778#   988
POPRT   1632#
PSHNEG  3600#  3676   4142
PSH      247    252    621   655   898    899  1266  1270  1276  1280  1623  1632  2327  2391
        2424   2428   2437  2546  2558   2589  2772  2789  2793  2800  2802  2827  2845  2916
        3660   3734   3744  3747  3845   3852  4024  4042
PUSHF     89    234    791  1168#  2670   3618  3705  3815  3837  3975  3992  4115  4123
QINLIN  4225#  4340   4395  4503
QINT      89   1530#  1616  2817
QINTA   1557   1579#
READY   4225#  4613
REALIO    5#     12   4237  4332  4385
REASON  4225#  4573
REPINI  4601#  4601
RND       89   3884#
RND1    3895   3922#
RNDX    3898   3930   3945#
ROUND    279    296    395   412#
ROUNDA   418    452#  1568
```

ACI	752														
ACRLF	46#	4663	4719	4751	4753	4780	4802								
ADC	385	477	481	2581											
ADD	400	473	976	2689	2691	2828	4439								
ADI	535	984	1032	2943	3727	3729	4435								
ANA	1581														
ANI	424	4242	4273												
CALL	166	192	200	233	237	248	253	275	292	625	631	637	643	652	
	656	680	790	800	819	990	1026	1038	1187	1351	1548	1551	1562	1615	
	1629	2339	2425	2548	2586	2590	2649	2672	2705	2719	2773	2782	2790	2797	
	2803	2809	2816	2819	2859	2882	2886	2952	3609	3615	3647	3655	3661	3683	
	3688	3704	3713	3724	3738	3741	3748	3751	3757	3812	3820	3823	3836	3841	
	3854	3858	3863	3900	3907	3910	3919	3922	3934	3961	3974	3983	3988	3991	
	3994	3999	4011	4019	4039	4114	4117	4122	4126	4129	4160	4170	4233	4322	
	4336	4339	4375	4391	4394	4405	4499	4582	4572	4585	4588	4594	4603		
CC	315	1567	1572	3675											
CMRGET	414	2346	2363	2394	4342	4397	4505								
CM	417	1556	2762	4141	4144										
CMA	230	493	998	1066	2345	2932	4437								
CMC	891														
CMP	1360	1364	1368	4359	4365										
CNC	4043														
CNZ	2913														
COMPAR	438	4562													
CP	2418	4025													
CPI	244	359	1606	2325	2331	2352	2357	2371	2377	2382	2385	2833	2905	2911	
	3719	4150	4343	4414	4506	4514	4519	4533							
CZ	2438	2853	3678												
DAD	380	750	924	1700	1705	1710	4455	4584							
DC	4641	4666	4723	4739	4755	4783	4806								
DCR	379	539	771	906	945	1250	1585	1714	2370	2559	2843	2850	2894	2898	
	3848	4363													
DCX	620	631	836	994	1362	1366	1370	1579	2336	2392	2877	2878	2903	3930	
	4363	4384	4459												
FSIGN	44#	604	676	813	997	1077	1115	1342	2747	3633	3886	4014	4022	4140	
IN	4238	4240	4271												
INR	231	283	450	452	454	456	630	822	823	905	1044	1070	1582	2380	
	2396	2405	2408	2429	2801	2812	2838	2865	2933	2937	4438				
INX	282	305	389	422	475	479	710	1180	1220	1222	1224	1226	1248	1249	
	1272	1273	1358	2755	2869	2873	2885	2892	2923	2946	2948	2950	3840	4355	
	4554														
JC	323	1034	1701	1711	2349	2395	3731	4152	4416	4503					
JM	900	2830	3894												
JMP	185	295	428	465	532	556	950	1019	1046	1104	2604	2676	2695	2794	
	3765	3867	3940	4073	4132	4369									
JNC	227	278	402	741	892	1707	2835	2879	2940	3721	4432	4456			
JNZ	340	361	773	938	947	1251	1715	2359	2399	2430	2895	4349	4360	4379	
	4418	4521													
JP	274	387	981	1016	2421	2758	2927	3650	4016						
JPE	605														
JPO	2786	2966													
JZ	219	284	394	713	814	968	987	1335	2328	2333	2354	2409	2758	2899	
	2907	2917	3634	3639	4244	4275	4345	4366	4399	4511	4516	4535			

LDA	215	928	1082	1609	3716	4147									
LDAX	1246														
LHLD	1169	1174													
LXI	179	269	397	413	489	610	634	640	692	697	701	706	747	942	
	972	1041	1085	1093	1123	1210	1232	1262	1338	1345	1402	1692	2366	2702	
	2726	2742	2823	2847	3608	3612	3735	3754	3760	3815	3897	3937	3958	4008	
	4034	4070	4155	4163	4167	4173	4238	4247	4255	4279	4297	4304	4310	4326	
	4333	4552	4372	4388	4402	4445	4496	4508	4524	4527	4547	4559	4581	4591	
	4597	4612													
MOV	212	251	301	302	304	306	308	310	312	342	343	345	346	352	
	353	354	355	356	381	383	384	386	390	391	392	401	412	420	
	423	427	472	474	476	478	480	482	492	494	496	498	499	501	
	502	504	505	507	527	528	529	537	544	546	547	549	550	552	
	553	555	613	622	684	695	696	709	739	740	761	762	764	765	
	767	768	770	772	781	782	783	784	827	832	837	864	867	868	
	869	875	878	879	882	885	886	904	915	917	918	920	921	923	
	925	927	935	966	971	977	980	986	993	1029	1037	1096	1097	1126	
	1129	1199	1200	1219	1221	1225	1225	1247	1265	1269	1274	1275	1279	1333	
	1343	1349	1359	1363	1367	1371	1530	1531	1532	1533	1555	1565	1580	1605	
	1614	1622	1624	1695	2342	2343	2344	2346	2404	2416	2579	2580	2582	2686	
	2694	2839	2866	2868	2870	2872	2874	2876	2891	2904	2947	2949	2951	3637	
	3644	3665	3764	3839	3925	3926	3927	3931	4158	4159	4358	4364	4408	4413	
	4575	4577	4578	4580											
MOVRI	57#	614	646	659	794	2776	2956	3707	3913	3977					
MVI	298	456	463	516	530	678	721	817	887	1083	1098	1101	1241	1559	
	1570	1618	1698	2717	2740	2753	2756	2840	2851	2863	2921	2925	2930	2935	
	3926	3932	4253	4259	4285	4291	4356	4552							
ORA	213	218	268	347	393	416	712	936	937	967	1000	1030	1334	1534	
	1696	3638	4023	4348	4378	4398	4409								
ORI	1267	1277	3645												
OUTCHR	42#														
PCHL	2969	3604	4620												
POP	198	199	210	211	240	241	252	628	629	655	674	675	751	778	
	800	807	898	899	902	903	946	999	1001	1270	1280	1374	1375	1576	
	1632	2391	2428	2436	2437	2558	2589	2601	2602	2603	2708	2709	2800	2827	
	2844	2862	2890	2893	2916	2952	2965	3618	3619	3658	3659	3664	3667	3671	
	3686	3687	3744	3745	3746	3763	3826	3845	3846	3847	3857	3866	3986	3987	
	3997	3998	4042	4120	4121	4448	4566								
POPR	54#	198	210	240	628	674	806	3618	3658	3686	3745	3846	3986	3997	
PUSH	247	621	623	624	700	704	705	716	746	873	874	941	1173	1178	
	1266	1276	1341	1547	1623	2327	2365	2415	2424	2546	2578	2583	2584	2585	
	2701	2729	2745	2761	2772	2793	2802	2845	2856	2857	2858	3642	3643	3653	
	3654	3660	3681	3682	3734	3747	3818	3819	3852	3851	3852	3853	3861	4024	
	4166	4387	4460												
PUSHFM	48#														
PUSHM	45#														
PUSHR	51#	623	3642	3653	3681	3850									
RAL	382	911	916	919	922	926	931	1067	1103	1566	1625				
RAR	545	548	551	554	735	760	763	766	769	907	978	1354	3666		
RET	376	483	588	779	785	995	1071	1130	1180	1205	1227	1254	1282	1356	
	1376	1577	1598	1633	1718	2442	2560	2953	4176						
RLC	2687	2688	2690												
RM	1350														

```
RNC          246    1612
RNZ          451     453     455     459    1045    1069    1361    1365    1369    1373    1583
RP          1116
RST          604     676     813     997    1077    1115    1342    2348    2363    2394    2747    3633    3886    4014
            4022    4140    4342    4397    4505    4562
RZ           214     405     540     677    1031    1344    1535    1697    2376    2379    2384    2390    2545    3849
            3903
SBB          303     307     311     500     503     506    1068    2871    2875    4579
SBI          880     884     889
SHLD         689    1196    1201    3668    3672    4250    4255    4261    4268    4282    4287    4293    4300    4307
            4314    4329    4449    4452    4530    4538    4541    4544    4555    4568    4600    4617
SPHL        4313    4567
STA          373     685     828     833     838     870     895     932    4319    4419    4423    4427    4440
STC          900    4015
SUB          226     300     497     620    1372    1561    2403    2417    2867    4576
SUI          357     521     876    2590    2692    2938    4430
SYNCHK      39# 
XCHG         232     236     688     720     771     865    1168    1179    1195    1204    1704    1706    2441    2715
            2884    4125    4382    4551    4571
XRA          299     344     372     426     495     538     866     975     979    1271    1281    1348    1355    1554
            2402    2716    2771    4318
XRI         1127
XTHL        1172    1177    2369    3603    4556
```

```
SPOOLER RUNTIME 14 SECONDS, 44 KCS, 402 DISK READS, 3 DISK WRITES, 110 PAGES
 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

 **END** USER MITS  [6000,6000] JOB  F4 SEQ. 205 DATE 27-AUG-75 07:36:23 MONITOR ALBUQUERQUE SCHOOLS 507A **END**

000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111
```